Approximating Weighted Shortest Paths on Polyhedral Surfaces [*†‡]

Mark Lanthier, Anil Maheshwari and Jörg-Rüdiger Sack
School of Computer Science
Carleton University
1125 Colonel By Drive
Ottawa, ON, CANADA K1S 5B6
E-mail: {lanthier,maheshwa,sack}@scs.carleton.ca

Abstract

Consider a simple polyhedron $\mathcal{P}$, possibly non-convex, composed of n triangular regions (faces), in which each region has an associated positive weight. The cost of travel through each region is the distance traveled times its weight. We present and experimentally study several algorithms to compute an approximate weighted geodesic shortest path, $\pi'(s,t)$, between two points s and t on the surface of $\mathcal{P}$. Our algorithms are simple, practical, less prone to numerical problems, adaptable to a wide spectrum of weight functions, and use only elementary data structures. An additional feature of our algorithms is that execution time and space utilization can be traded off for accuracy; likewise, a sequence of approximate shortest paths for a given pair of points can be computed with increasing accuracy (and execution time) if desired. Dynamic changes to the polyhedron (removal, insertions of vertices or faces) are easily handled. The key step in these algorithms is the construction of a graph by introducing Steiner points on the edges of the given polyhedron and compute a shortest path in the resulting graph using Dijkstra's algorithm. Different strategies for Steiner point placement are examined. Our experimental results obtained on Triangular Irregular Networks (TINs) modeling terrains in Geographical Information Systems (GIS) show that a constant number of Steiner points per edge suffice to obtain high-quality approximate shortest paths. The time complexity of these algorithms for TINs (obtained using real data and randomly generated data) which we experimentally investigated is $\mathcal{O}(n \log n)$.

Our analysis bounds the approximate shortest path cost, $\pi'(s,t)$, by $\pi(s,t) + w_{max} \cdot |l_e|$, where $\pi(s,t)$ denotes the geodesic shortest path between s and t on the boundary of $\mathcal{P}$, l_e is the longest edge and w_{max} is the maximum weight of the faces of $\mathcal{P}$, respectively. The worst case time complexity is bounded by $\mathcal{O}(n^5)$. We present an alternate algorithm, using graph spanners, that runs in $\mathcal{O}(n^3 \log n)$ worst case time and reports an approximate path such that $\pi'(s,t) \leq \beta(\pi(s,t) + w_{max} \cdot |l_e|)$, where $\beta > 1$ is a constant. Already, for planar subdivisions, the best known algorithm for computing exact geodesic weighted shortest path runs in $\mathcal{O}(n^8 \log n)$ time and using $\mathcal{O}(n^4)$ space, due to Mitchell and Papadimitriou [12]. We are not aware of any adequately documented algorithm for computing approximate weighted shortest paths.

1 Introduction

Shortest path problems are among the fundamental problems studied in computational geometry. In this paper, we consider the problem of computing a shortest cost path between two points s and t on a polyhedral surface $\mathcal{P}$. The surface is composed of triangular regions (faces) in which each region has an associated positive weight indicating the cost of travel in that region. This problem arises in numerous applications in areas such as robotics, traffic control, search and rescue, water flow analysis, road design, navigation, routing, geographical information systems. Most of these applications demand simple and efficient algorithms to compute approximate shortest paths as opposed to a complex algorithm that computes an exact path. Polyhedra arising in these applications approximate real surfaces and thus an approximate path will typically suffice. Our interest is also motivated by our research and development on a parallel system for GIS and spatial modeling.

*Research supported in part by ALMERCO Inc. & NSERC.
†For the full version of this paper see [9].
‡See [10] for an accompanying video.

Research work on computing a shortest geodesic path on unweighted polyhedral surfaces began with the special case of convex polyhedra [14, 17]. Approximate computations of Euclidean paths on unweighted polyhedra have recently been reported by [1, 7]. The computation of Euclidean shortest paths on non-convex polyhedra has been investigated by [3, 11, 15, 19]; currently, the best known algorithm due to Chen and Han [3] runs in $\mathcal{O}(n^2)$ time. * The best known algorithm for the query problem, where both source and destination are unknown, requires $\mathcal{O}(n^6 m^{1+\delta})$, for $\delta > 0$ and $1 \leq m \leq n^2$, to answer queries in $\mathcal{O}((n/m^{1/4}) \log n)$ time [2].

Mitchell and Papadimitriou [12] introduced the "weighted region problem" and presented an algorithm that computes a shortest weighted cost path between two points in a planar subdivision; it requires $\mathcal{O}(n^8 \log n)$ time in the worst case. They state that their algorithm applies to non-convex polyhedral terrains with modifications. (See the survey paper by Mitchell and Suri [13] for several results on shortest path problems.)

Motivated by the practical importance of these problems and very high complexities for computing exact shortest paths, we investigate algorithms for computing approximated shortest paths. In this paper we propose several simple and practical algorithms (schemes) to compute an approximated weighted shortest path $\pi'(s,t)$ between two points s and t on the surface of a (possibly, non-convex) polyhedron $\mathcal{P}$. The accuracy of the approximation varies with the length of the longest edge l_e of $\mathcal{P}$ and the maximum weight w_{max} of the faces of $\mathcal{P}$. We present two variations on the algorithm. The first ensures that $|\pi'(s,t)| \leq |\pi(s,t)| + w_{max} \cdot |l_e|$. The second, based on graph spanners, ensures that $|\pi'(s,t)| \leq \beta(|\pi(s,t)| + w_{max} \cdot |l_e|)$ for some fixed constant $\beta > 1$. Our experimental results indicate that the schemes presented here perform very well in practice and guarantee excellent results for polyhedra in which $|\pi(s,t)| >> |l_e|$. The typical running time for our algorithms is $\mathcal{O}(n \log n)$ although the worst case running times for our algorithms are $\mathcal{O}(n^5)$ and $\mathcal{O}(n^3 \log n)$, respectively.

Our methodology for computing approximate shortest paths in unweighted TINs involves computing the sequence of faces through which the approximate path passes and then constructing the path by performing a "sleeve computation". In some cases the set of faces intersected by our approximate shortest path is identical to that of the actual shortest path, and hence the "sleeve computation" reports the actual shortest path.

*Very recently, Varadarajan and Agarwal [19] propose an algorithm that computes an unweighted approximated path which is at most 13 times the actual path length and requires $\mathcal{O}(n^{\frac{5}{3}} \log^{\frac{5}{3}} n)$ time. They provide another algorithm that requires $\mathcal{O}(n^{\frac{8}{5}} \log^{\frac{8}{5}} n)$ time and produces a path that is at most 15 times the optimal.

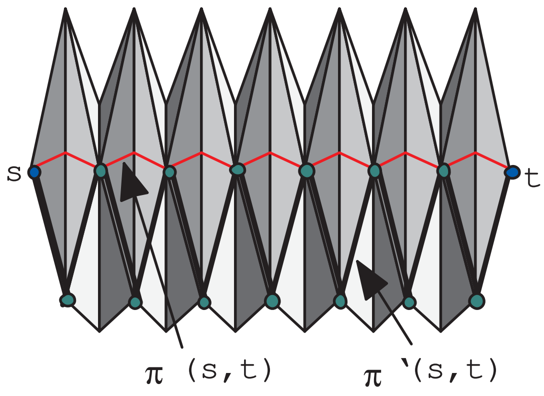

Figure 1: A worst case example

2 Shortest Path Approximations

2.1 A Simple Approach

Given a polyhedron $\mathcal{P}$, we could compute a graph G (with edge weights) as follows. The vertices in G correspond to the vertices of $\mathcal{P}$ and there is an edge between two vertices in G if the corresponding vertices in $\mathcal{P}$ are connected by an edge.

For simplicity, assume that the source, s, and the target, t, are vertices of $\mathcal{P}$. Let $\pi(s,t)$ be composed of a sequence of k adjacent straight line segments $s_1, s_2, ..., s_k$. The cost of a segment $s_j, 1 \leq j \leq k$ passing through face $f_i, 1 \leq i \leq n$ is assumed to be $w_{fi} \cdot |s_j|$. Similarly, an approximated path is denoted by $\pi'(s,t)$ with cost $|\pi'(s,t)|$ and segments $s'_1, s'_2, ..., s'_{k'}$.

$\pi'(s,t)$ can be computed using Dijkstra's algorithm [5]. This scheme confines the path to traveling on edges of $\mathcal{P}$, the quality of the approximation depends on the given triangulation, which could be bad in the worst case (see Figure 1). The approximated path $\pi'(s,t)$ with k' links may have length $k' \cdot |l_e|$ and this could be much larger than $|\pi(s,t)|$.

2.2 Our Approximation Algorithms

We improve on the previous scheme by introducing Steiner points on the edges of the polyhedron. The set of vertices of G will now consist of vertices and Steiner points of $\mathcal{P}$. We designed several strategies for placing vertices and edges in the graph G; this distinguishes the schemes examined here.

2.2.1 Fixed Scheme

We add m Steiner points evenly along each edge of $\mathcal{P}$, for some positive integer m. For each face $f_i, 1 \leq i \leq n$ of $\mathcal{P}$, compute a subgraph G_i as follows. The Steiner points, along with the original vertices of f_i, become vertices of G_i. Connect a vertex pair v_a, v_b of G_i to form an edge of G_i if and only if a) v_a and v_b represent Steiner points that lie on different edges of f_i or b) v_a and v_b represent Steiner points that are adjacent on the same edge of f_i. The weight on a graph edge is the Euclidean distance between v_a and v_b times the weight of f_i. Denote G_i as a complete graph although some edges are missing (i.e., those connecting vertices of non-adjacent Steiner points of the same polyhedral edge). A graph G is then computed by forming the union of all $G_i, 1 \leq i \leq n$.

The approximated shortest path in $\mathcal{P}$ is computed by first determining a shortest path in G using Dijkstra's algorithm and then transforming this path to a corresponding path in $\mathcal{P}$. It can easily be shown that all edges of G lie on the surface of $\mathcal{P}$. Hence, any path in G (i.e., our approximation) must also be a path on the surface of $\mathcal{P}$.

In order to analyze the cost of an approximated path, we first consider how each segment of a shortest path is approximated. If we compute a path $\pi'(s,t)$ in G that passes through the same sequence of faces as $\pi(s,t)$, then the following claim is made:

Claim 1 We can approximate a segment $s_j, 1 \leq j \leq k$ of $\pi(s,t)$ passing through face $f_i, 1 \leq i \leq n$ with an edge s'_j of G_i such that $w_{fi} \cdot |s'_j| \leq w_{fi} \cdot |s_j| + w_{fi} \cdot \frac{|l_e|}{m+1}$, where m is the number of Steiner points added to each edge of the face f_i.

Proof: Each edge is divided into $m+1$ intervals which have length at most $\frac{|l_e|}{m+1}$. From the properties of weighted paths as described by Mitchell and Papadimitriou [12], it follows that s_j either crosses f_i completely as shown in Figure 2 or it travels along an edge of f_i (i.e., they show that the weighted shortest path obeys Snell's law of refraction and bends only at the edges of $\mathcal{P}$). Assume that s_j begins at a point a inside an interval of an edge of f_i and ends at the point b inside an interval of another edge (possibly the same one) of f_i. Without loss of generality assume that s'_j begins at an interval endpoint (i.e., a Steiner point), say c, that is closest to a and ends at an interval endpoint, say d, which is closest to b. Now if c and d are on different edges we know that there is a Steiner edge joining them. If they lie on the same edge, then there exists a series of adjacent collinear Steiner edges joining them, and we view these collinear edges as a single segment s'_j.

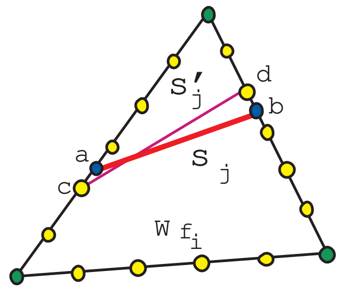

Figure 2: A weighted shortest path segment s_j that crosses a face.

In Figure 2, the triangle inequality ensures that $|s'_j| \leq |\overline{ca}| + |s_j| + |\overline{bd}|$. Since we chose the closer interval endpoints, then $|\overline{ca}| \leq \frac{l_e}{2(m+1)}$ and $|\overline{bd}| \leq \frac{l_e}{2(m+1)}$. Hence

$$|s'_j| \leq |s_j| + \frac{l_e}{m+1}. \qquad (1)$$

Now, multiplying by w_{f_i} we have

$$w_{f_i} \cdot |s'_j| \leq w_{f_i} \cdot |s_j| + w_{f_i} \cdot \frac{|l_e|}{m+1}. \qquad (2)$$

In the case where a segment of the path travels along an edge of $\mathcal{P}$, we know that $|\overline{ca}|$ and $|\overline{bd}|$ have the same bounds as in 2. It is easily shown that equations 1 and 2 still hold. The proof also applies to the first and last segments of $\pi(s,t)$ in which s and t may be internal to faces.□

Lemma 1 There exists an approximated path $\pi'(s,t)$ in $\mathcal{P}$ such that $|\pi'(s,t)| \leq |\pi(s,t)| + \frac{|l_e|}{m+1} \cdot k \cdot w_{max}$.

Proof: Let $\pi'(s,t)$ be an approximated path with k segments that passes through the same sequence of faces as $\pi(s,t)$. We write: $|\pi(s,t)| = \sum_{i=1}^{k}(w_{f_{s_i}} \cdot |s_i|)$ and $|\pi'(s,t)| = \sum_{i=1}^{k}(w_{f_{s_i}} \cdot |s'_i|)$ where $w_{f_{s_i}}$ is the weight of the face that s_i passes through. By applying the results of Claim 1 to each segment of $\pi(s,t)$ we have: $\sum_{i=1}^{k}(w_{f_{s_i}} \cdot |s'_i|) \leq \sum_{i=1}^{k}(w_{f_{s_i}} \cdot |s_i| + w_{f_{s_i}} \cdot \frac{|l_e|}{m+1})$.

We can rewrite this as

$$|\pi'(s,t)| \leq |\pi(s,t)| + \frac{|l_e|}{m+1} \cdot \sum_{i=1}^{k}(w_{f_{s_i}}).$$

Since $w_{f_{s_i}} \leq w_{max}$ for all f_{s_i} by definition, then

$$|\pi'(s,t)| \leq |\pi(s,t)| + \frac{|l_e|}{m+1} \cdot k \cdot w_{max}. \qquad (3)$$

□

Since a shortest weighted path on $\mathcal{P}$ may cross an edge $\mathcal{O}(n)$ times, we obtain the following theorem:

Theorem 1 Using the fixed scheme, we can compute an approximation $\pi'(s,t)$ of the weighted shortest path $\pi(s,t)$ between two points s and t on a polyhedral surface $\mathcal{P}$ such that $|\pi'(s,t)| \leq |\pi(s,t)| + w_{max} \cdot |l_e|$, where l_e is the longest edge of $\mathcal{P}$ and w_{max} is the maximum weight of any face of $\mathcal{P}$. Moreover, we can compute this path in $\mathcal{O}(n^5)$ time.

Proof: We know (from Mitchell and Papadimitriou [12]) that a shortest weighted path on $\mathcal{P}$ may cross an edge $\mathcal{O}(n)$ times. Hence, a weighted shortest path may have $\mathcal{O}(n^2)$ segments. In Lemma 1, set $k = n^2$ and $m = n^2$, then we obtain:

$$|\pi'(s,t)| \leq |\pi(s,t)| + w_{max} \cdot |l_e|. \tag{4}$$

Using the fixed scheme, each edge contributes $\mathcal{O}(n^2)$ graph vertices and each face contributes $\mathcal{O}(n^4)$ graph edges. Thus, the algorithm requires $\mathcal{O}(n^3)$ space for graph vertices and $\mathcal{O}(n^5)$ space for graph edges. Dijkstra's shortest path algorithm takes $\mathcal{O}(n^5)$ time on this graph, using a fibonacci heap. $\square$

2.2.2 Interval Scheme

In the fixed scheme, we made an assumption in our analysis that each edge crossed by the shortest path was of length $|l_e|$. In practice there are many edges of $\mathcal{P}$ with small length compared to $|l_e|$ (we have examined edge-length histograms for all of our data). This means that the addition of m Steiner points to the smaller edges resulted in very small intervals with length much less than $\frac{|l_e|}{m+1}$.

We can improve the fixed scheme by forcing the intervals between adjacent Steiner points on an edge to be of length $\frac{|l_e|}{m+1}$. As a result, we can typically reduce the number of Steiner points added per edge considerably.

Since the maximum length of an interval is at most $\frac{|l_e|}{m+1}$, the proofs of Claim 1 and Lemma 1 still apply and hence Theorem 1 holds for the interval scheme. Although, the worst case analysis is the same for both schemes, the reduction of Steiner points has the advantage of reducing the number of graph vertices and edges that are created and processed by the graph shortest path algorithm.

2.2.3 Spanner Schemes

The time complexities of the previous two schemes can be further improved as described next; though the approximation achieved is not as good. Intuitively, we should be able to eliminate some edges joining Steiner points without a drastic reduction in our approximation factor. We eliminate these edges by using the notion of a *spanner*. Let $G_i, 1 \leq i \leq n$ be the complete graph formed by applying the edge decomposition scheme on a face f_i of $\mathcal{P}$ with m Steiner points per edge. G_i has $3(m+1)$ vertices and $3(m+1)^2$ edges (including those along the face boundaries). We construct a *β-spanner* of G_i and call it G'_i. The vertices of G'_i are the vertices of G_i which we sort in CCW order around any interior point of f_i to form a sorted list $V_{G'_i}$.

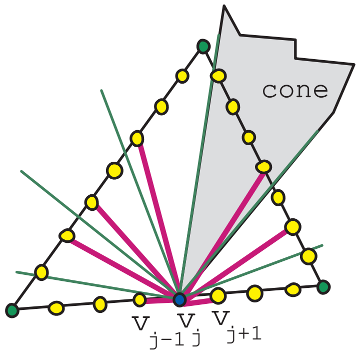

Figure 3: The spanner edges added from a vertex v_j with $\theta = 30°$.

Let $\mathcal{C}$ be the set of 2-d cones with apex at $v_j, 1 \leq j \leq 3(m+1)$ and the conical angle $\theta = \frac{\pi}{\rho}$ for $\rho > 4$. For each vertex v_j perform a radial sweep of the vertices $v_r, (j+1)\text{mod}(3(m+1)) \leq r \leq (j-1+3(m+1))\text{mod}(3(m+1))$. During this sweep compute the vertex v_{min} that has minimal distance to v_j in each of the ρ cones and add an edge to G'_i connecting v_j and v_{min} (see Figure 3).

Results of Clarkson [4], ensure that G'_i is a β-spanner for G_i where $\beta = \frac{1}{cos\theta - sin\theta}$. Since G'_i is a β-spanner with $\mathcal{O}(m)$ vertices, it has $\mathcal{O}(m)$ edges. We create a similar spanner for each face of $\mathcal{P}$ individually and then merge each $G'_i, 1 \leq i \leq n$ to form the union G'. The approximate shortest path $\pi'(s,t)$ is computed by computing a shortest path in G'.

Claim 2 A segment $s_j, 1 \leq j \leq k$ of $\pi(s,t)$ passing through f_i can be approximated by a path p_j in G'_i such that $w_{f_i} \cdot |p_j| \leq \beta \cdot w_{f_i} \cdot |s_j| + \beta \cdot w_{f_i} \cdot \frac{|l_e|}{m+1}$, where m is the number of Steiner points added per edge.

Proof: From Claim 1 it follows that $w_{f_i} \cdot |s'_j| \leq w_{f_i} \cdot |s_j| + w_{f_i} \cdot \frac{|l_e|}{m+1}$ for some segment s'_j in G_i. From the definition of G'_i, it follows that s'_j can be approximated by a path p_j in G'_i such that $|p_j| \leq \beta \cdot |s'_j|$. By substituting this into the result of Claim 1,

$$w_{f_i} \cdot \frac{|p_j|}{\beta} \leq w_{f_i} \cdot |s'_j| \leq w_{f_i} \cdot |s_j| + w_{f_i} \cdot \frac{|l_e|}{m+1} \quad (5)$$

Hence,

$$w_{f_i} \cdot |p_j| \leq \beta \cdot w_{f_i} \cdot |s_j| + \beta \cdot w_{f_i} \cdot \frac{|l_e|}{m+1}. \quad (6)$$

□

Lemma 2 By applying the β-spanner scheme to $\mathcal{P}$ using m Steiner points per edge, an approximate path $\pi'(s,t)$ is obtained for which $|\pi'(s,t)| \leq \beta(|\pi(s,t)| + \frac{|l_e|}{m+1} \cdot k \cdot w_{max})$.

Proof: The proof uses the results of Claim 2 and the techniques of Lemma 1. Let $\pi'(s,t)$ be the approximated path with k subpaths $p_1, p_2, ..., p_k$ that passes through the same faces as $s_1, s_2, ..., s_k$ of $\pi(s,t)$ respectively. From the definition of the path cost it follows that

$|\pi(s,t)| = \sum_{i=1}^{k}(w_{f_{s_i}} \cdot |s_i|)$ and

$|\pi'(s,t)| = \sum_{i=1}^{k}(w_{f_{s_i}} \cdot |p_i|)$.

By applying the results of Claim 2 to each segment of $\pi(s,t)$ we have

$\sum_{i=1}^{k}(w_{f_{s_i}} \cdot |p_i|) \leq \sum_{i=1}^{k}(\beta \cdot w_{f_{s_i}} \cdot |s_i| + \beta \cdot w_{f_{s_i}} \cdot \frac{|l_e|}{m+1})$.

Rewrite this as

$|\pi'(s,t)| \leq \beta(|\pi(s,t)| + \frac{|l_e|}{m+1} \cdot \sum_{i=1}^{k}(w_{f_{s_i}}))$

Since $w_{f_{s_i}} \leq w_{max}$ for all f_{s_i} by definition, then

$$|\pi'(s,t)| \leq \beta(|\pi(s,t)| + \frac{|l_e|}{m+1} \cdot k \cdot w_{max}). \quad (7)$$

□

Theorem 2 An approximate weighted shortest path $\pi'(s,t)$ between two points s and t on a polyhedron of n faces can be computed in $\mathcal{O}(n^3 \log n)$ time such that $|\pi'(s,t)| \leq \beta(|\pi(s,t)| + w_{max} \cdot |l_e|)$, $\beta > 1$, where l_e is the longest edge of $\mathcal{P}$ and w_{max} is the maximum weight of any face of $\mathcal{P}$.

Proof: The correctness follows from Lemma 2. We apply the β-spanner scheme to obtain G'_i for each face f_i of $\mathcal{P}$ where $1 \leq i \leq n$. Using $\mathcal{O}(n^2)$ Steiner points per edge, each subgraph G'_i contains $\mathcal{O}(n^2)$ graph vertices and each vertex has a constant (depending upon β) degree since G'_i is a spanner. Each G'_i is computed by applying a radial sweep to each vertex in order to add its incident edges. Using a simple approach, this sweep could take $\mathcal{O}(n^2)$ time per vertex. However, since there are a constant number of cones ($\approx 2\rho$), we can divide $\mathcal{O}(n^2)$ vertices into an equal number of partitions. Since these vertices are sorted radially, we can apply a binary search for each of the cones to obtain the partitions in $\mathcal{O}(\log n)$ time. For each partition, we can again apply a binary search to obtain the closest vertex (v_{min}) in $\mathcal{O}(\log n)$ time as well. Since there are a constant number of partitions, each of the $\mathcal{O}(n^2)$ vertices can be processed in $\mathcal{O}(\log n)$ time and hence G'_i can be computed in $\mathcal{O}(n^2 \log n)$ time. Since there are n faces, G' can be computed in $\mathcal{O}(n^3 \log n)$ time. A shortest path in G' can be computed by using Dijkstra's algorithm (with a Fibonacci heap) and it runs in $\mathcal{O}(n^3 \log n)$ time. No further improvement on the graph construction is stated as the cost matches that of Dijkstra's shortest path algorithm. □

2.2.4 Sleeve Based Schemes

The scheme discussed next applies only to unweighted polyhedra. We first compute the approximate shortest path $\pi'(s,t)$ either using the fixed or the interval scheme. We then determine a sleeve by unfolding the faces along the edge sequence of $\pi'(s,t)$ and compute the shortest path that lies within this sleeve. We have implemented an algorithm similar to the algorithm of Guibas and Hershberger [6] and have applied this to our approximated path. (We omit the details of our sleeve computation due to space constraints from this version.) Section 3 shows that these approximated paths are much more accurate with this additional computation at a negligible increase in execution time. In most of our cases, the edge sequence of the approximated path is identical to that of an exact shortest path. The sleeve computation then produces the exact shortest path.

There is no efficient algorithm for computing shortest paths in weighted sleeves. For the weighted case however, we can perform a second approximation based on the outcome of the first. To do this, we select a *buffer* of faces from $\mathcal{P}$ to be used in the computation of the second approximation by selecting all faces that were "passed through" by the first approximated path. If this approximated path passed through a vertex, we add all incident faces of this vertex to the buffer. We then apply the approximation scheme on the buffer faces with an increased number of Steiner points per edge. As a result, we obtain a refined path. The refinement can be repeated as many times as desired.

2.3 Implementation Related Issues

A Variation of Dijkstra's Algorithm: In place of using Dijkstra's shortest path graph algorithm, we implemented a known variation of the algorithm. In this scheme we associate an additional weight to each vertex, namely its Euclidean distance to the target vertex. In this algorithm, at any iteration, the least cost vertex which we choose is the one which minimizes the sum of its cost from the source vertex plus the Euclidean distance to the target vertex, over all possible vertices.

Space Complexity: In applications, the space requirements for algorithms on TINs are often important. To address this, in any of our schemes, we can store the graph as it pertains to Steiner points implicitly. In an iteration of Dijkstra's algorithm, adjacency information can be computed on the fly with little penalty.

Numerical Issues: Suri [18] points out that the Chen and Han algorithm [3] based on unfolding is sensitive to numerical problems. This is due to the fact that 3D rotations are performed and errors accumulate along the paths and geometric structures computed. In our schemes the paths go through vertices or Steiner points (with the exception of the variation using as final step a sleeve computation); thus reducing the accumulation of numerical errors.

3 Experimental Results

A natural subclass of non-convex polyhedra is a *Triangular Irregular Network* or TIN which has triangular faces. A TIN is often constructed from a triangulated point set in the plane in which each point is assigned a height. In Geographic Information Systems, Cartography and related areas, shortest path problems arise frequently on terrains which are often modeled using TINs as shown in Figure 4. Due to their practical relevance, and in the context of our R&D [8], our experimental results are for TINs although the algorithms presented here also apply to any non-convex polyhedra.

One of the main difficulties in presenting experimental results is to choose *benchmark* TINs. It is conceivable that different TIN characteristics could affect the performance of an algorithm. We have attempted to accommodate different characteristics by performing our tests with TINs that have different sizes (i.e. number of faces), height characteristics (i.e. smooth or spiky (modeled by accentuating the heights)), and data sources (i.e. random or sampled from Digital Elevation Models (DEM)). Table 1 shows the attributes of the TINs that we've tested. TINs with stretched heights were created by multiplying their heights by five.

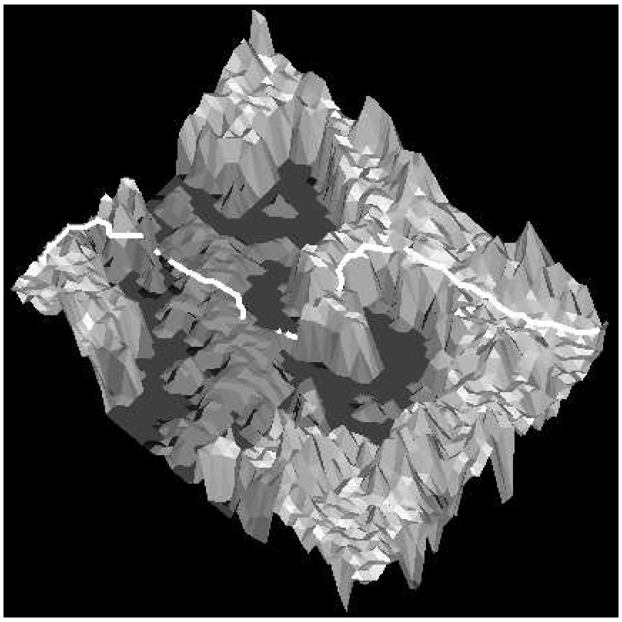

Figure 4: A weighted shortest path on a terrain in which traveling on water is expensive.

FACES	STRETCHED HEIGHTS	DATA SOURCE
1012	NO	DEM
1012	YES	DEM
5000	NO	RANDOM
5000	YES	RANDOM
10082	NO	DEM
10082	YES	DEM

Table 1: The various TINs and their attributes.

For each TIN, we computed a set of 100 random vertex pairs. We then tested each of the approximation schemes as shown in Table 2. We give the id of each scheme as they appear in the graphs. For each test, we computed the path cost between each of the 100 vertex pairs and then obtained an "average path cost" for these pairs. The tests were performed in iterations based on the number of Steiner points per edge. Each scheme was tested for both weighted and non-weighted scenarios (with the exception of the sleeve computations which were only computed in the unweighted case; a second approximation using a buffer was applied in the weighted case).

For the weighted domain, we use the same TINs and

LEGEND ID	SCHEME	SLEEVE or BUFFER
INT	INTERVAL	NO
FIX	FIXED	NO
INTSLV	INTERVAL	YES
FIXSLV	FIXED	YES

Table 2: The different approximation schemes.

we assign a weight to each face as the slope of the face. Thus, steeper faces have higher weight. Each edge of the TIN is given weight equal to the minimum of its adjacent faces. We determined experimentally that the spanner schemes provide slightly worse approximations and do not provide adequate improvement in running time; we omit the graphs here. Our analysis for the approximation, bounds the number of Steiner points by $\mathcal{O}(n^2)$ per edge in the worst case, this is far from the values required in our test suites using typical TINs.

3.1 Path Accuracy

We first ran tests on a terrain where the weight was homogeneous throughout all faces, i.e., we examined the Euclidean shortest path problem. The graphs in Figure 6 depict the results of these tests for variations of our approximation schemes. We can see that by using only a small number of Steiner points per edge, the approximated path length quickly converges to the actual path length (as computed using Chen and Han algorithm [3]). The graph shows that only a small number of Steiner points (i.e., 6) per edge suffice to obtain close-to-optimal approximations. Furthermore, the path accuracy observed is far better than the theoretical bound derived. The graphs also illustrate that the additional sleeve computation helps to obtain even more accurate approximations. Therefore, the best of our unweighted schemes is the interval scheme with the sleeve computation (IntSlv).

The graphs on the right of Figure 6 represent the approximations we obtained on the TINs with the heights of the vertices multiplied by a factor of 5. The graphs show that the results remain very good in that the approximate path length converges after only a few Steiner points per edge are added. We do see however, that the convergence is not as quick. This is mainly due to the fact that Steiner points are placed further apart along the now longer edges. Therefore it requires more Steiner points to reduce the interval size to that of the flatter TIN. The interval scheme performs better than the fixed scheme, since interval scheme favors placement of Steiner points on longer edges and longer edges are more likely to be crossed by the set of our random paths.

3.2 Computation Time

Since we are adding only a constant number of Steiner points on the average per edge, the running time of our algorithms is $\mathcal{O}(n \log n)$. In general, the algorithms' running times depend on the number of Steiner edges since we are evoking Dijkstra's graph shortest path algorithm.

The graphs of Figure 7 depict the running time for our 10,082 face TIN. We can see that our algorithms are substantially faster than Chen and Han[3]. The main reason is that our algorithms do not require any complex data structures nor do they perform expensive computations (i.e., 3D rotation and unfoldings). We precompute the graph G and then perform a search for each query; a query is for a pair (source, destination), and we measure the time it takes to answer a query. From the graphs, we can see that the time required for the additional sleeve computation is negligible.

3.3 Weighted Paths

One problem in determining the accuracy of the algorithm in the weighted scenario is that we do not have an implementation of any algorithm that determines the actual shortest weighted path. The graphs of Figure 8 show the accuracy obtained through experimentation. As with the unweighted case, our path costs converge to some value after only a few Steiner points were added per edge. We therefore conjecture, that the cost of the paths converge similar to the unweighted scenario. From the graphs we can see that the second approximation based on the buffer technique provides a similar increase in accuracy as with the unweighted sleeve computations.

Since our algorithm is the same for unweighted and weighted scenarios, we obtained almost identical running time as shown in Figure 9. We can see however, that the second approximation resulted in a significant increase in computation time with respect to the increase shown for the unweighted sleeve computations. This increase is mainly due to the construction of a new refined graph which is necessary for each query.

4 Conclusion and Future Work

Shortest path problems belong to a class of geometric problems that are fundamental and of significant practical relevance. While realistic shortest path problems frequently arise in applications where the cost of travel is not uniform over the domain, the time, space and implementation complexities of existing algorithms even for the planar case are extremely large which motivates our study of approximation algorithms. Our experimental results show that high-quality approximations can be obtained with very good run-times. More precisely, we have provided empirical results showing that typical terrain data requires only a few (constant) Steiner points per edge. This reduces the running time to $\mathcal{O}(n \log n)$ in practice which is orders of magnitude smaller than the best known exact shortest path algorithm. The solutions are simple and of practical value.

We also theoretically establish bounds on the approximation quality and give worst-case bounds on the run-time of our algorithms.

For the unweighted scenario, we compared our accuracy to that of Chen and Han [3] and gave results indicating that our algorithm performs up to 50 times faster with a mimimum observed speedup of 14 times and produces nearly identical path results. We claim that our algorithm is efficient w.r.t. accuracy versus running time and is simple to implement. Our algorithm is of particular interest also for the case of queries with unknown source and destination.

Currently, we are investigating scenarios that involve other realistic weights taking into consideration physical properties of vehicles (see Figure 5). We are also working on other schemes and a parallel implementation of our algorithms.

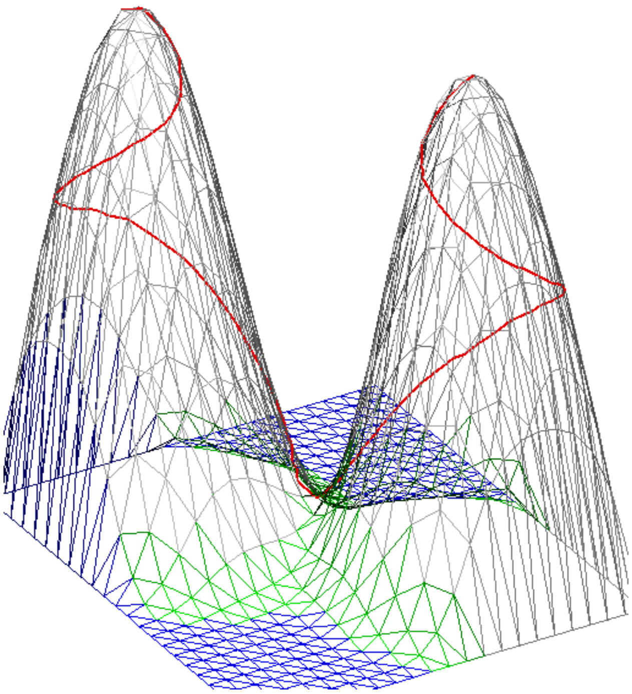

Figure 5: Shortest weighted path taking into consideration the maximum slope a vehicle can travel, as well as the turning angle.

Acknowledgments: We would like to thank Doron Nussbaum for useful discussions.

References

[1] S. Har-Peled, M. Sharir, and K.R. Varadarajan, "Approximating Shortest Paths on a Convex Polytope in Three Dimensions", *Proc. 12th Annual Symp. on Computational Geometry*, Philadelphia, PA, 1996, pp. 329-338.

[2] P.K. Agarwal, B. Aronov, J. O'Rourke, and C. Schevon, "Star Unfolding of a Polytope with Applications", to appear in *SIAM Journal on Computing*, 1997.

[3] J. Chen and Y. Han, "Shortest Paths on a Polyhedron", *Proc. of the 6th ACM Symposium on Computational Geometry*, 1990, pp. 360-369.

[4] K.L. Clarkson, "Approximation algorithms for shortest path motion planning", *Proc. 19th Annual ACM Symp. Theory of Computing*, 1987, pp. 56-65.

[5] E.W. Dijkstra, "A Note on Two Problems in Connection with Graphs", *Numerical Mathematics 1*, 1959, pp.269-271.

[6] L. Guibas and J. Hershberger, "Optimal Shortest Path Queries in a Simple Polygons", *Proc. of the 3rd ACM Symposium on Computational Geometry*, 1987, pp. 50-63.

[7] J. Hershberger and S. Suri, "Practical Methods for Approximating Shortest Paths on a Convex Polytope in $\Re^3$", *Proc. of the 6th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1995, pp. 447-456.

[8] D. Hutchinson, M. Lanthier, A. Maheshwari, D. Nussbaum, D. Roytenberg, J.-R. Sack, "Parallel Neighbourhood Modeling", *Proc. of the 4th ACM Workshop on Advances in Geographic Information Systems*,Minnesota, 1996, pp. 25-34.

[9] M. Lanthier, A. Maheshwari and J.-R. Sack, "Approximating Weighted Shortest Paths on Polyhedral Surfaces", Tech. Report TR-96-32, Carleton University, Dec. 96.

[10] M. Lanthier, A. Maheshwari and J.-R. Sack, "Approximating Weighted Shortest Paths on Polyhedral Surfaces", video in this proceedings, *13th Annual ACM Symposium on Computational Geometry*, June 1997.

[11] J.S.B. Mitchell, D.M. Mount and C.H. Papadimitriou, "The Discrete Geodesic Problem", *SIAM Journal of Computing*, **16**, August 1987, pp. 647-668.

[12] J.S.B. Mitchell and C.H. Papadimitriou, "The Weighted Region Problem: Finding Shortest Paths Through a Weighted Planar Subdivision", *Journal of the ACM*, **38**, January 1991, pp. 18-73.

[13] J.S.B. Mitchell and S. Suri, "A Survey of Computational Geometry", M.O. Ball et al. Eds., *Handbooks in Operations Research and Management Science*, Elsevier Science B.V., Vol. 7, Chapter 7, 1995, pp. 439-458.

[14] D.M. Mount, "On Finding Shortest Paths on Convex Polyhedra", Tech. Report 1495, Department of Computer Science, University of Maryland, Baltimore, MD, 1985.

[15] J. O'Rourke, S. Suri and H. Booth, "Shortest Paths on Polyhedral Surfaces", extended abstract, Dept. Electrical Engineering and Computer Science, Johns Hopkins University, Baltimore, Maryland, September 1984.

[16] N.C. Rowe, and R.S. Ross, "Optimal Grid-Free Path Planning Across Arbitrarily Contoured Terrain with Anisotropic Friction and Gravity Effects", *IEEE Transactions on Robotics and Automation*, Vol. 6, No. 5, October 1990, pp. 540-553.

[17] M. Sharir and A. Schorr, "On Shortest Paths in Polyhedral Spaces", *SIAM Journal of Computing*, **15**, 1986, pp. 193-215.

[18] S. Suri, personal communication, 1996.

[19] K.R. Varadarajan and P.K. Agarwal, "Approximating Shortest Paths on a Polyhedron", unpublished report, November 1996.

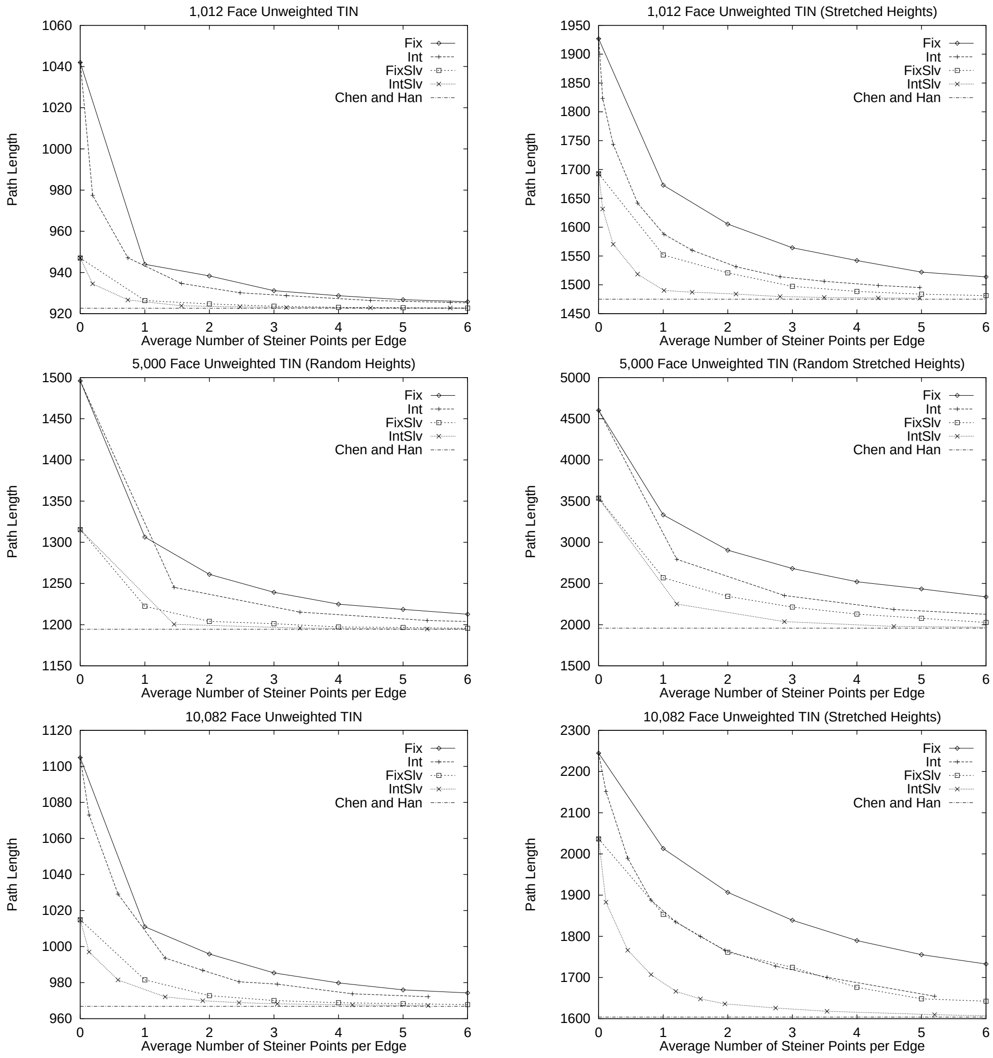

Figure 6: Graphs showing avg. path length.

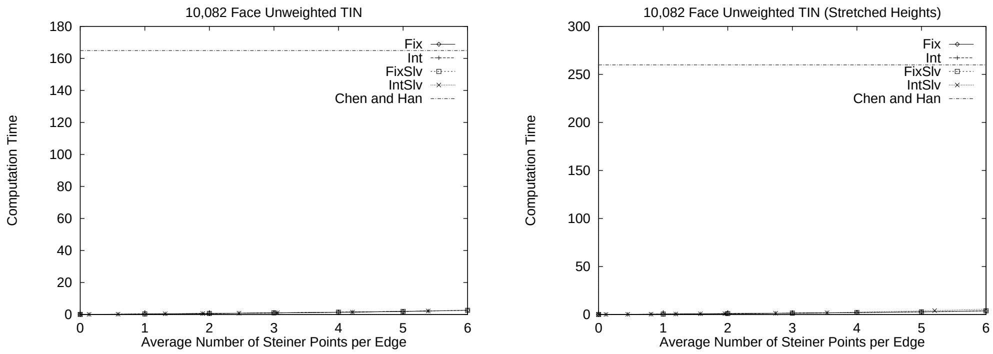

Figure 7: Graphs showing avg. computation time.

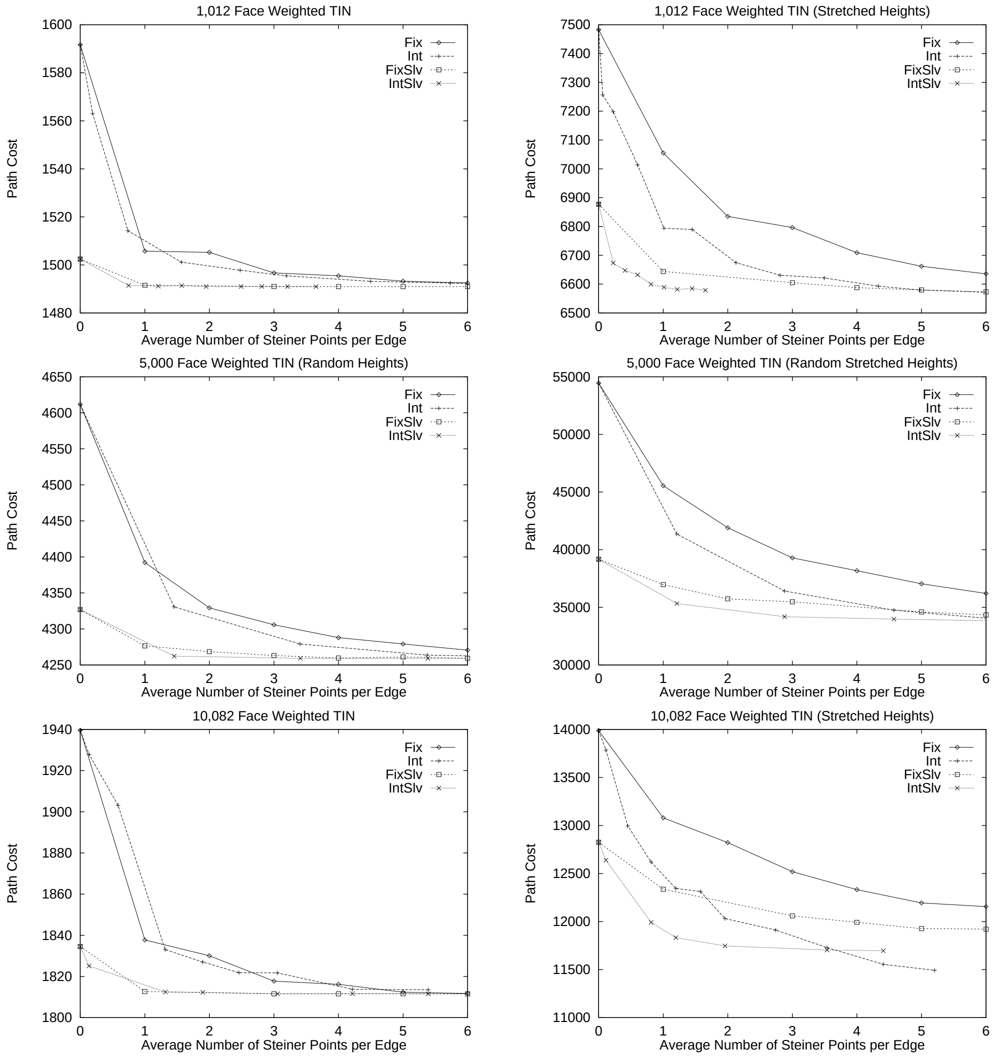

Figure 8: Graphs showing avg. path cost.

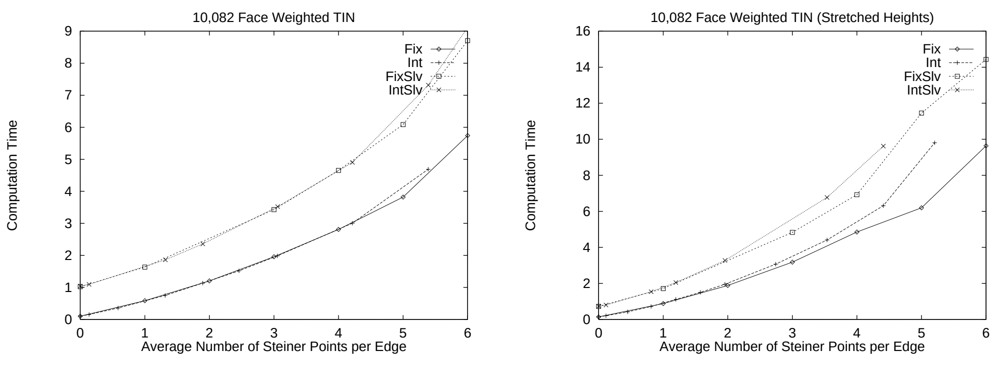

Figure 9: Graphs showing avg. computation time.