An Experimental Study of Weighted k-Link Shortest Path Algorithms

Ovidiu Daescu[1], Joseph S. B. Mitchell[2], Simeon Ntafos[1], James D. Palmer[1], and Chee K. Yap[3]

[1] Department of Computer Science, The University of Texas at Dallas, Richardson, TX 75080, USA, {daescu,ntafos,jdp011100}@utdallas.edu. †
[2] Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, NY 11794, USA. jsbm@ams.sunysb.edu ‡
[3] Department of Computer Science, Courant Institute, New York University, New York, NY 10012, USA. yap@cs.nyu.edu §

Abstract: We consider the problem of computing a minimum-weight polygonal path between two points in a weighted polygonal subdivision, subject to the constraint that the path have few segments (*links*). We give an algorithm that generates paths of weighted length at most $(1+\epsilon)$ times the weight of a minimum-cost k-link path, for any fixed $\epsilon > 0$, while using at most $2k-1$ links. This is an improvement over the previous $(1+\epsilon)$-approximation algorithm, which used at most $5k-2$ links. Further, we have implemented our new algorithm and we have conducted a performance study of these algorithms (old and new) on a variety of real-world and synthetic data, comparing not only the efficiency but also the quality of paths generated using these algorithms. We also consider the implications of these results on the practical usage of these algorithms.

1 Introduction

Consider a mobile robot that moves within an environment that is partitioned into regions, each having an associated *weight*, which represents the cost per unit distance for travel in the region. Furthermore, assume that there is a cost associated with each turn that the robot makes. Given an environment specified by a weighted polygonal subdivision, R, and given a positive integer k, our goal is to compute a minimum-cost polygonal path for the robot, from point a to point b, subject to the constraint that the path have at most a specified number, $k-1$, of turns, and therefore at most k links (edges).

† Daescu's research is supported by NSF grant CCF-0430366.
‡ J. Mitchell is partially supported by the U.S.-Israel Binational Science Foundation (2000160), NASA (NAG2-1620), NSF (CCR-0098172, ACI-0328930, CCF-0431030), and Metron Aviation.
§ Yap's work is supported by NSF Grant CCF-0430836.

In a weighted subdivision the distance between two points a and b within the same region $R_i \in R$ is defined as the product of the weight w_i of R_i and the Euclidean length $|ab|$ of the line segment ab. For a path p (not necessarily polygonal), the portion of p that is contained within some region $R_i \in R$ has its length defined as the product of the Euclidean length of $p \cap R_i$ and the weight w_i. The length $||p||$ of the path p is the sum of the lengths over each region it intersects. The weight of a path that follows an edge $e_{i,j}$ that forms the boundary between two regions R_i and R_j is defined as $\min\{w_i, w_j\}$. We assume R is triangulated and has n vertices in general position.

1.1 Related Work

For a survey of optimal path algorithms in geometry, see [14, 15].

Minimum-cost paths in weighted subdivisions were first studied algorithmically by Mitchell and Papadimitriou [16], who give a $(1+\epsilon)$-approximation algorithm to compute optimal paths that runs in polynomial time (logarithmic in $1/\epsilon$). Alternative solution methods for the weighted region problem (WRP) are based on discretizing edges of the subdivision, placing Steiner points judiciously, and interconnecting them to form a discretization graph, $G(V, E)$, which is then searched for a shortest path. The first experimental studies of the weighted region problem ([11,13]) were based on implementations of such algorithms. Much of the subsequent WRP work has focused on the placement of Steiner points either on edges [1, 2, 21] or on face bisectors [3], with clever techniques to speed the computation of shortest paths in the discretization graph [3, 21]. None of these algorithms for the WRP permit one to bound the number of links/turns in the produced path.

The special case of the 1-link minimum-weight path between two regions in a weighted subdivision, often called the optimal "link" or "penetration" problem, was first studied in [5, 6]. In [5] the optimal link problem is reduced to $O(n^2)$ subproblems each of which minimizes a two-variable function $f(x, y)$ over a convex domain D, where $f(x, y)$ is given as a sum of $O(n)$ fractional terms. An optimal link between two regions must pass through a vertex in the subdivision [7], a property that has been exploited to create efficient approximation algorithms based on a *prune and search* approach and a *sum of fractionals* approach [9].

The first algorithms to approximate k-link shortest paths in weighted regions, with guaranteed approximation bounds, are proposed in [9]. Two different algorithms for approximating k-link shortest path are proposed based on the computation of *exact* optimal links or *approximate* optimal links. Their solutions produce approximation paths that are within an $(1+\epsilon)$-factor from optimal (ϵ-approximation paths for short) and have $5k-2$ links and $14k$ links, respectively. As in the algorithms in [3, 12, 21], a discretization graph $G(V, E)$ is constructed from Steiner points placed on edges in the subdivision. The construction of G differs in two important ways from discretizations for the shortest path problem. The first difference arises in the placement of Steiner

points. A vertex-vicinity (see Section 1.3) is chosen small enough so that no line can intersect three vertex-vicinities. The second difference is in the definition of G. Nodes in V correspond to Steiner edges (not points) and vertices in the subdivision. Edges in E correspond to optimal links between the nodes in V.

The problem of computing optimal k-link paths in weighted regions has been studied in air traffic management applications, where the weights correspond to traversability or risk factors associated with hazardous weather or congestion, and the turn constraint corresponds to the need to bound the human factors complexity of flight trajectories, both from the pilot's perspective and from the air traffic control perspective. The algorithms implemented for this application in Krozel et al. [10] rely on searching grids, using Bellman-Ford methods, but do not provide guarantees of solution quality.

The problem of computing shortest k-link paths in *unweighted* settings (e.g., in simple polygons or polygonal domains) has been studied by Piatko et al. [4, 17, 19].

1.2 Our Contribution

In this paper we describe two new techniques for approximating k-link shortest paths in weighted regions, we implement a solver with a suite of algorithms (new and old), and we perform an experimental investigation:

- We prove that by shrinking the vertex-vicinity by a constant factor of μ, which depends on some parameters of the subdivision, we can find a $(2k-1)$-link path p formed from exact optimal links and turning only on edges, such that $||p|| \leq (1+2\epsilon)||p_k||$ where p_k is an optimal k-link path from s to t.
- We show that using approximate optimal links we can find a $(2k-1)$-link path p formed from approximate optimal links and turning only on edges such that $||p|| \leq (1+7\epsilon)||p_k||$ where p_k is an optimal k-link path from s to t.
- We implement our new methods, as well as those previously proposed. We have built an extensive software system, "k-LinkSolver", which is available for public use.
- We conduct the first experimental investigation of these k-link path solutions. We compare and contrast the run-time performance and solution quality of our new algorithms with those described in [9], and consider the implications of our results on the practical usage of these algorithms. This is the first intensive study on the real-world performance of any of these algorithms.

We refer to a path as being an *ϵ-good approximation* if its weighted length is within a $(1+\epsilon)$-factor of the weight of a corresponding optimal path. That is, p is an ϵ-good approximation of a path p_k if $||p|| \leq (1+\epsilon)||p_k||$. A *$C\epsilon$-good*

approximation refers to a path whose weighted length is within a $(1 + C\epsilon)$-factor of the weight of an optimal path where C is a constant. By letting $\epsilon' = \epsilon/C$ it is clear that a $C\epsilon$-good approximation forms an ϵ'-good approximation. That is, any $C\epsilon$-good approximation trivially forms an ϵ-good approximation. From this point forward, we will generally refer to paths as ϵ-good disregarding the constant C to mean this and we will state results as being $C\epsilon$-good or within a $(1+C\epsilon)$-factor to acknowledge the specific constant C required. Thus, the two solutions presented here form ϵ-good approximations.

1.3 Definitions and Notations

We begin by describing the terminology and discretization scheme used here. Whenever possible, our definitions and notations are similar to those in [9]. Let E_R be the set of edges that bound the weighted regions of the weighted subdivision R. Let $E_R(v)$ be the set of boundary edges incident to a point v and let $d(v)$ be the minimum Euclidean distance between v and the edges in $E_R \setminus E_R(v)$. For each edge $e \in E_R$, let $d(e) = \sup_{v \in e} d(v)$. Let $v(e)$ be a point on e such that $d(v(e)) = d(e)$. The cell formed by the regions incident to a vertex v is called the *vertex-cell* of v and is denoted by $C(v)$ (C_i for short for a vertex v_i).

For each vertex of R in each possible vertex triplet formed by the vertices in R we compute the minimum distance to the line supporting the opposite edge. Let γ_i be the minimum such distance obtained from a triplet containing the vertex v_i, and let $\gamma = \min\{\gamma_i/2 \mid i = 1, 2, \ldots, n\}$. Using a result in [8], γ can be found in $O(n^2 \log n)$ time.

For each vertex v of R, let $r(v) = \min(\epsilon d(v)/c, \gamma)$ where c is an appropriately chosen constant (see Lemma 2). The disk of radius $r(v)$ centered at a vertex v defines the *vertex-vicinity* $S(v)$ of v (or S_i for short for a vertex v_i). We refer to $r(v)$ as the *vertex-vicinity radius* for v. Note that the choice for $r(v)$ ensures no line can cross more than two vertex-vicinities. This depends on the non-degeneracy assumption that no three vertices of the subdivision are collinear.

We now describe how the Steiner points on an edge $e = \overline{v_1 v_2}$ are chosen. Each vertex v_i, where $i = 1, 2$, has a vertex-vicinity S_i of radius $r(v_i)$ and the Steiner points $v_{i,1}, \ldots, v_{i,j_i}$ are placed on e such that $|v_i v_{i,1}| = r(v_i)$ and $|v_{i,m} v_{i,m+1}| \leq \epsilon d(v_{i,m})$, $m = 1, 2, \ldots, j_i - 1$ (where equality holds in all cases except possibly when $m = j_i - 1$). The value of j_i is such that $v_{i,j_i} = v(e)$. Let δ be the maximum number of Steiner points placed on an edge of the subdivision. The line segment formed by two adjacent Steiner points $v_{i,m}$ and $v_{i,m+1}$ is called a *Steiner edge*. The pairing of any two Steiner edges forms a quadrilateral shape called a *Steiner strip*. The shape could be degenerate if the Steiner edges are on the same boundary edge or share a vertex.

If a path in R is restricted to turn only on edges it is said to be *edge-restricted* otherwise the path is said to be *edge-unrestricted*. If a link that

makes up a polygonal path follows an edge of the subdivision (rather than crossing a face) it is said to be *edge-crawling*.

A key subproblem in approximating k-link shortest paths is that of finding "good" 1-link paths (links for short). We refer to two different kinds of good links. An exact optimal link is one that has been solved such that no error exists. An approximate optimal link is within a $(1+\epsilon)$-factor from an optimal link. Finally, for a path p_k, we use $|p_k|$ to denote its Euclidean length and $||p_k||$ to denote its weighted length.

2 Approximating k-link Shortest Paths

We recall a key theorem from [9]:

Theorem 1. *Given two points s and t of R, there exists a path p between s and t with at most $(2k-1)$-links, that turns only on the edges of the subdivision and such that the weighted length of p is at most that of a k-link shortest path p_k from s to t.*

The edges added to transform an edge-unrestricted path to an edge-restricted path can be thought of as cutting the "corners" off the edge-unrestricted path against the edges of the subdivision. We call such links *corner-cutting* links.

In [9] it is shown how to construct "normalized" paths that either completely or partially avoid vertex-vicinities. The penalty of such an approach is often a higher constant multiplier bounding the number of links. In this section we propose an alternative that decreases the number of links in an approximating path to $(2k-1)$ at the cost of increased complexity in the discretization graph.

We do this by decreasing the size of the vertex-vicinity radius to $r(v) = \min(\mu\epsilon d(v)/c, \gamma)$ where $\mu = w_{min}/w_{max}$ and c is an appropriately chosen constant (see Lemma 2). The discretization graph $G(V, E)$ is defined similarly to [9] but with a few subtle changes. Nodes in V still correspond to Steiner edges and vertices in the subdivision but also include what we will refer to as *interior Steiner edges*. An interior Steiner edge is the edge between a vertex and the first Steiner point on an edge incident to the vertex. Edges in E then correspond to optimal links between pairs of nodes in V.

Lemma 1. *A k-link shortest path, p_k, completely contained in a vertex-vicinity has a maximum weighted length of $2\epsilon w_{min} d(v)/c$.*

Proof. Let l be a link from the start of the path p_k to the end of the path p_k (see Fig. 1). Clearly, if $||p_k|| > ||l||$ then p_k is not an optimal path. Therefore, $||p_k|| \leq ||l||$. The length of l is constrained by the size of the vertex-vicinity and thus, $|l| < 2\epsilon\mu d(v)/c$ or $||l|| < 2\epsilon w_{min} d(v)/c$. And finally, $||p_k|| < 2\epsilon w_{min} d(v)/c$. We may also conclude that only one link is required for a path p'_k to approximate a path p_k such that $||p'_k|| < 2\epsilon w_{min} d(v)/c$.

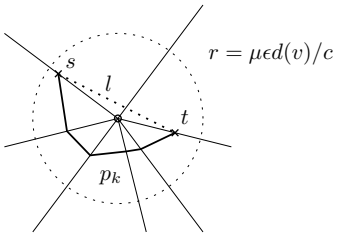


Fig. 1. p_k has a maximum weighted length of $2\epsilon w_{min} d(v)/c$ where c is an appropriately chosen constant.

In this paper, we assume the end points of a path p_k are separated by a distance $\Omega(d(v_i))$ for any vertex-vicinity S_i through which p_k passes. If $|p_k| \ll d(v_i)$, then p_k is a trivial path.

Lemma 2. *A k-link path, p_k, that turns on edges can be approximated by a 2ϵ-good $(2k-1)$-link path made up of k optimal links and $k-1$ connecting links.*

Proof. Let the path p_k be divided into j subpaths, $p_{k_1}, \ldots, p_{k_j}$, as defined below (see Fig. 2). Let $p_{k_0} = \emptyset$. If $i = 1$ then subpath p_{k_i} begins where p_k begins otherwise subpath p_{k_i} begins where $p_{k_{i-1}}$ ends. With $p_{k_{i-1}}$ known, the subpath p_{k_i} is defined as follows. Consider the first vertex-vicinity S_i that p_k crosses after $p_{k_{i-1}}$.

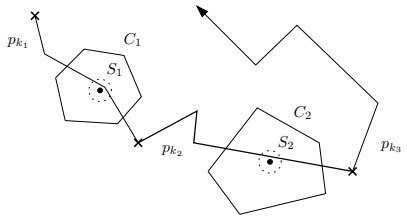


Fig. 2. Three subpaths, p_{k_1}, p_{k_2} and p_{k_3}.

Case 1: If p_k crosses the boundary of the vertex-cell C_i before entering S_i then let the subpath p_{k_i} terminate at the first turn after p_k exits S_i. If no such turn exists, or p_k does not exit S_i, then let p_{k_i} end where p_k ends.

Case 2: If p_k does not cross the boundary of C_i before entering S_i, then let the subpath p_{k_i} terminate at the first turn after p_k exits C_i. If no turn exists after p_k exits C_i then let p_{k_i} end where p_k ends. If p_k does not exit C_i, then we let $p_{k_{i-1}}$ end where p_k ends.

A subpath p_{k_i} so constructed may intersect no more than four vertex-vicinities and $|p_{k_i}| = \Omega(d(v))$ for any vertex-vicinity v crossed by p_{k_i} (see below for a more precise lower bound). Four vertex-vicinities may be crossed if Case 2 is applied once in a situation similar to the last example in Fig. 3

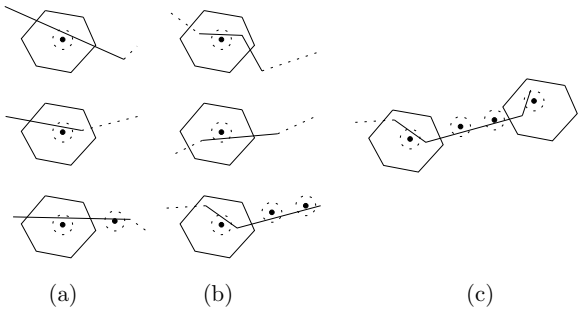


Fig. 3. Several examples of (a) Case 1 and (b) Case 2 described in Lemma 2 and (c) where Case 2 is applied twice on a single subpath.

(b) and then again for a small link that crosses a vertex-vicinity as illustrated in Fig. 3 (c).

Note that $(1+\epsilon)||p_k|| = \sum_{i=1}^{j}(1+\epsilon)||p_{k_i}||$. Let p'_k be a path that approximates p_k with k optimal links and $k-1$ connecting links. We prove p'_k can be constructed such that $||p'_k|| < (1+2\epsilon)||p_k||$ by proving $||p'_{k_i}|| < (1+2\epsilon)||p_{k_i}||$.

First consider a subpath π of p_{k_i} that does not cross a vertex-vicinity and its approximating subpath π'. Such a path π is made up of one or more links, $l_1, \ldots, l_{\xi_i}$. Then $||\pi|| = \sum_{j=1}^{\xi_i} ||l_j||$ and by extension, $(1+\epsilon)||\pi|| = \sum_{j=1}^{\xi_i}(1+\epsilon)||l_j||$. Let e_1 and e_2 be the Steiner edges that a link $l_j \in \pi$ originates and terminates at respectively. Let l_j^* be an optimal link between e_1 and e_2. Clearly, $||l_j^*|| \leq ||l_j||$. Consider the contribution of the last region that l_j crosses before connecting to l_{j+1}. Let this region be R_1 with corresponding weight w_1. Let $d_1 = |R_1 \cap l_j|$. Since in general the endpoints of l_j^* and l_{j+1}^* on e_2 are distinct, we need to add a single link, l_{m_j}, to connect l_j^* and l_{j+1}^* on e_2 (l_{m_j} is an edge-crawling connecting link). The link l_{m_j} can have length no greater than that of the Steiner edge on which it lays.

We have $|l_{m_j}| \leq \epsilon d(v_2)$, with v_2 an endpoint of the Steiner edge e_2. By definition, $d(v_2) \leq d_1$. Therefore, $|l_{m_j}| \leq \epsilon d_1$. The contribution l_{m_j} makes to approximate l_j is no more than $\epsilon w_1 d_1$, i.e., $d_1 w_1 + ||l_{m_j}|| \leq (1+\epsilon) w_1 d_1$. Using l_j^* and l_{m_j} to approximate l_j we have $||l_j^*|| + ||l_{m_j}|| \leq (1+\epsilon)||l_j||$, since $||l_j^*|| \leq ||l_j||$. It then follows that $||\pi'|| = \sum_{j=1}^{\xi_i}(||l_j^*|| + ||l_{m_j}||) < \sum_{j=1}^{\xi_i}(1+\epsilon)||l_j|| = (1+\epsilon)||\pi||$. Equivalently, $\sum_{j=1}^{\xi_i} ||l_{m_j}|| < \epsilon||\pi||$.

Next consider what happens when p_{k_i} crosses a vertex-vicinity $S(v)$ of vertex v. From Lemma 1, if p_{k_i} consists of several links contained in $S(v)$, they can be replaced by a single link, of weighted length no more than $2\epsilon w_{min} d(v)$.

Let l_s be a link in the path p_{k_i} that enters $S(v)$ and let l_t be a link that exits $S(v)$ (see Fig. 4). Let l_s^* be an optimal link that begins and ends on the same Steiner edges as l_s. Let l_t^* be an optimal link that begins and ends on the same Steiner edges as l_t. Clearly, $||l_s^*|| \leq ||l_s||$ and $||l_t^*|| \leq ||l_t||$. Let l_m be a connecting link (not necessarily edge-crawling) that connects l_s^* to

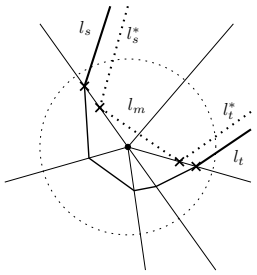


Fig. 4. Approximating path construction inside a vertex-vicinity.

l_t^*. We need to bound l_m in terms of p_{k_i}. We can find a lower bound for the unweighted length of p_{k_i} by using $d(v)$ and the vertex-vicinity radius of $S(v)$, i.e., $(1 - \epsilon\mu/c)d(v) < |p_{k_i}|$ and thus $d(v) < c|p_{k_i}|/(c - 1/2)$. The link l_m can achieve an unweighted value of no more than twice the radius of the vertex-vicinity, thus l_m's maximum weighted value is bounded by $||l_m|| \leq 2\epsilon w_{min} d(v)/c < 2\epsilon w_{min}|p_{k_i}|/(c - 1/2) \leq 2\epsilon||p_{k_i}||/(c - 1/2)$. Also recall that up to four vertex-vicinities may exist in a single subpath p_{k_i}. If we let $c = 17$, then $4||l_m|| < \frac{\epsilon}{2}||p_{k_i}||$.

Combining this result with the result we found for edge-crawling links l_{m_j} not inside the vertex-vicinity, we have a path p'_{k_i} with $||p'_{k_i}|| < (1 + 3\epsilon/2)||p_{k_i}|| < (1+2\epsilon)||p_{k_i}||$. That is, a 2ϵ-good subpath p'_{k_i} can be guaranteed. And thus, a 2ϵ-good path p'_k can also be guaranteed.

Theorem 2. *Given two points s and t of R, a k-link shortest path between s and t can be approximated with a 2ϵ-good $(2k - 1)$-link path that turns on edges.*

Proof. Consider a link l_i^* on the k-link shortest path and refer to Fig. 5. If l_i^* ends on an edge of the subdivision then Lemma 2 applies and it can be replaced in the approximating path by one optimal link and one connecting link. If l_i^* ends inside a face of R then Theorem 1 applies, adding a corner-cutting link that crosses the face to connect l_i^* to l_{i+1}^*. No additional edge-crawling connecting link is required because the corner-cutting link crosses only one face and therefore its length is adequately captured in the discretization.

This new fact that links which cross a single face do not require additional connecting links was not considered in [9]. Applying this result, we find the $5k - 2$ link and $14k$ link results for optimal links and approximate optimal links are improved to $3k - 2$ links and $8k$ links respectively.

When computing the discretization graph $G(V, E)$, defined earlier, the number of vertices in V is clearly bounded by the number of Steiner points $O(\delta n)$ and the number of edges in E is $O((\delta n)^2)$. Computing a single edge in G corresponds to solving a 1-link shortest path problem for a specific subproblem. Let this time be $T_h(n)$. Thus, the time to compute G is $O((\delta n)^2 T_h(n))$. Once G is constructed, we can use dynamic programming to find a k-link shortest path in G in $O(k(\delta n)^2)$ time.

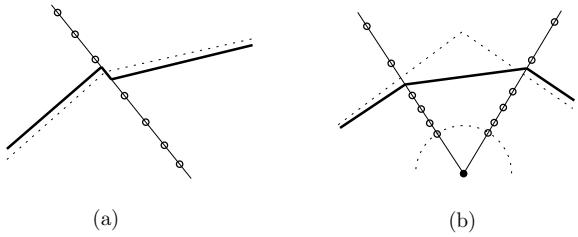


Fig. 5. a) Joining two optimal links with an edge-crawling connecting link (Case 1) and (b) joining two links not constrained to edges of the subdivision with a corner-cutting link (Case 2).

3 Approximate Optimal Links

In practice, optimal links can be time consuming to compute. Approximate optimal links are links that take advantage of the discretization scheme already described to find a single link within a $(1+\epsilon)$ factor of optimal. Such links were introduced in [9] but unfortunately the normalization process required several more links than it did for optimal links. In this section we show that approximate optimal links can be used with our new technique while still maintaining an approximating path with no more than $2k-1$ links.

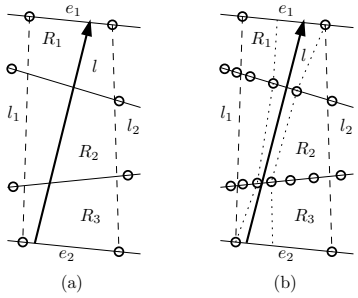


Fig. 6. A Steiner strip formed by lines l_1 and l_2 may intersect edges that are more coarsely (a) or more finely (b) sampled. In (b) the dotted lines describe an hourglass defined by Steiner points.

Consider an optimal link l between two edges e_1 and e_2. Many Steiner points may be captured inside the Steiner strip formed by e_1 and e_2. See Fig. 6. These points define a set of *hourglass* shaped regions, that represent the space over which a link can be translated and rotated without passing a Steiner point. An optimal link clearly must lie in one such hourglass. It has been shown in [9] that an arbitrary link computed for the same hourglass differs from an optimal link by only a factor of $1+2\epsilon$. Here we extend this result to include optimal links that cross or turn inside a vertex-vicinity.

Theorem 3. *Given two points s and t of R, a k-link shortest path between s and t can be approximated with a 7ϵ-good $(2k-1)$-link path made up of k approximate optimal links connected by $k-1$ connecting links.*

Proof. We divide the approximating path p'_k introduced earlier into j subpaths, $p'_{k_1}, \ldots, p'_{k_j}$, as already described in Lemma 2. We seek to bound the error between a subpath p''_{k_i} made of approximate optimal links and a subpath p'_{k_i} made of exact optimal links.

Consider a subpath p'_{k_i}. Such a path is made up of one or more optimal links, $l'_1, \ldots, l'_{\xi_i}$, and one or more small connecting links (but no more than $\xi_i - 1$). The connecting links will remain constrained to the size of the Steiner edge on which they lay or to twice the radius of a vertex-vicinity. Therefore, we are only concerned with how much l'_j and l''_j differ in length. In what follows we show $||l''_j||$ differs from $||l'_j||$ by at most a $(1+\epsilon)$ factor.

The weighted length of a single link, l'_j, intersecting m regions, $R_1, \ldots, R_m$, is given by $\sum_{i=1}^{m} w_i d_i$. We must consider the contribution of each segment to guarantee that the length of an approximating link l''_j is within a $(1+\epsilon)$-factor of the length of l'_j.

First consider a link l'_j that does not cross a vertex-vicinity. Let e_1 and e_2 be the Steiner edges that l'_j originates and terminates at respectively. Let V_S be the set of Steiner points contained in the Steiner strip formed from e_1 and e_2. V_S defines $O((\delta n)^2)$ hourglasses where δ is the number of Steiner points placed on an edge. Let the weighted length of an arbitrary line segment l''_j that passes through a particular hourglass approximate the length of the optimal line segment l'_j that passes through the same hourglass. Clearly each term that makes up the description of l''_j can vary from l'_j by at most the weighted lengths of the Steiner edges s_i and s_{i+1} corresponding to that term, i.e., $||l''_j|| = \sum_{i=1}^{m} w_i * d_i(l''_j) \leq \sum_{i=1}^{m} (w_i * d_i(l'_j) + w_i|s_i| + w_i|s_{i+1}|)$. By definition $|s_i| \leq \epsilon d_i(l')$ and $|s_{i+1}| \leq \epsilon d_i(l')$. Thus, $||l''|| = \sum_{i=1}^{m} w_i * d_i(l'') \leq (1 + 2\epsilon) \sum_{i=1}^{m} w_i d_i(l') = (1+2\epsilon)||l'||$.

The optimal link for the Steiner edges e_1 and e_2 is captured by one of the $O((\delta n)^2)$ hourglasses. The length of a representative link for one hourglass can be found in time proportional to the number of regions intersected by the link, and thus in $O(n)$ time. By taking the link of smallest weighted length over all hourglasses we have a link which approximates the optimal link for the Steiner strip within a factor of $(1+2\epsilon)$. The overall computation of this link takes $O(n(\delta n)^2)$ time.

Next, assume l'_j crosses a vertex-vicinity $S(v)$. Note that one or more links in p_{k_i} may pass through the vertex-vicinity. Links completely contained in the vertex-vicinity are in fact not captured at all, but we have shown the maximum amount of error induced by such links in Lemma 1. We focus our attention on two links in particular, the link entering and the link exiting the vertex-vicinity (which may be one in the same link).

As a link l'_j enters the vertex-vicinity $S(v)$, it must first pass a Steiner edge before it passes an internal Steiner edge. An arbitrary link l''_j that falls in the

same hourglass as l'_j is captured by a Steiner strip with an internal Steiner edge. Clearly the term associated with the description of l'_j in this strip can vary by at most $||s_i|| + \epsilon w_{min} d(v)/c$, where s_i is as above. If $\epsilon w_{min} d(v)/c > \epsilon d_i(l'_j)$ then the $(1 + 2\epsilon)$ bound we just discovered earlier does not hold. It is necessary to consider the error in our approximation that is induced by using internal Steiner edges not per link but per subpath.

The additional error associated with l''_j entering the vertex vicinity and crossing an internal Steiner edge is $\epsilon w_{min} d(v)/c$. Since the link l''_j may also exit the vertex-vicinity by crossing another Steiner strip formed from one Steiner edge and one internal Steiner edge, l''_j can vary by twice as much. The weighted length inside the vertex vicinity is also captured by $2\epsilon w_{min} d(v)/c$ where c is an appropriately chosen constant and thus crossing a single vertex-vicinity may cost $4\epsilon w_{min} d(v)/c$. Since a subpath p'_{k_i} may cross four vertex-vicinities, the maximum cost per subpath is $16\epsilon w_{min} d(v)/c < 16\epsilon w_{min} |p'_{k_i}|/(c - 1/2) < 16\epsilon ||p'_{k_i}||/(c - 1/2)$ and choosing $c = 17$ we have that $16\epsilon ||p'_{k_i}||/(17 - 1/2) < \epsilon ||p'_{k_i}||$.

That is, p''_{k_i} differs from p'_{k_i} by at most a factor of ϵ due to error from crossing internal Steiner edges. Adding this to the error incurred by crossing arbitrary Steiner edges, p''_{k_i} differs from p'_{k_i} by at most a factor of 3ϵ. Then, the difference between p''_{k_i} and p_{k_i} is $(1 + 3\epsilon)(1 + 3\epsilon/2) < 1 + 7\epsilon$; p''_{k_i} is a 7ϵ-good approximation of p_{k_i} and p''_k is a 7ϵ-good approximation of p_k.

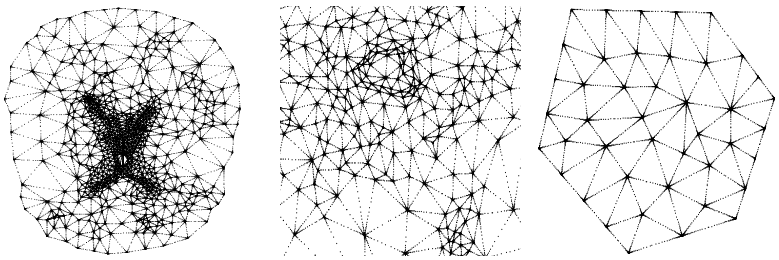
Substituting the link computation time $O(n(\delta n)^2)$ for $T_h(n)$ in the previous section we find the time to compute the discretization graph $G(V, E)$ is $O(n(\delta n)^4)$. As before, once G is constructed, we can use dynamic programming to find a k-link shortest path in G in $O(k(\delta n)^2)$ time.

4 Experiments

We have implemented the algorithms described in this paper as well as the algorithms described in [9] as part of a C++ application we call "k-LinkSolver". We refer to the k-link approximation algorithms in [9] using either optimal links or approximate optimal links as K-Link-Opt and K-Link-Approx, respectively, or K-Link collectively. We will refer to the new approximations based on either optimal links or approximate optimal links presented in this paper as K-Link-Mu-Opt and K-Link-Mu-Approx, respectively, or K-Link-Mu collectively.

We also define a new heuristic. Instead of solving for either an approximate or an optimal link between two Steiner edges we simply solve for an arbitrary link within a Steiner strip. When applied to the original discretization scheme we will call this algorithm K-Link-Heur and when applied to the discretization scheme used in this paper we call the technique K-Link-Mu-Heur. This heuristic makes no optimality guarantees but when ϵ is very small we might expect that an arbitrary link between two Steiner edges is very close in value to an optimal link between those two edges.

The k-LinkSolver solves for a k-link path in three stages: (1) discretization, (2) link generation and (3) path finding. The first and second stages do not depend on the start and end points of the path while the third stage does. If one is interested in finding several k-link paths in a single subdivision the first two stages can be thought of as preprocessing steps and the third stage can be executed as many times as required.



(a) (b) (c)

Fig. 7. A discretization of (a) MRI-DATA, (b) TOPO-DATA, and (c) TRI-DATA.

We use three different data sets in these experiments. The first is a traced MRI scan from the Visible Human Project [18] which emphasizes structure features of the brain (MRI-DATA). The second is a topological map of Santiago Peak from Big Bend National Park, Texas (TOPO-DATA). The third is a simple subdivision generated using Shewchuk's triangulator [20], Triangle, with specific quality and area guarantees (TRI-DATA). The discretization of MRI-DATA, TOPO-DATA, and TRI-DATA is shown in Fig. 7.

4.1 Discretization Performance

The first stage, discretization, is the most efficient. $O(n\delta)$ Steiner points are generated in $O(n\delta)$ time, with $\delta = O(1/\epsilon)$ (the hidden constant in the $O(\cdot)$ depends on some parameters of R, such as the maximum edge length and μ. Each Steiner point can be computed in constant time and each Steiner point becomes part of the discretization graph, making the discretization process inherently output sensitive.

Since Steiner points are associated (through edges) to nodes in the discretization graph, the number of Steiner points generated directly affects the number of links that must be calculated in the link generation phase and the number of links that must be considered in the path finding stage. Ideally we would like to generate as few Steiner points as possible that will make a specific $(1 + \epsilon)$ approximation guarantee.

Unlike K-LINK, K-LINK-MU depends on the ratio w_{min}/w_{max}. In our first series of tests, we fix ϵ to 0.5 and then vary μ to gauge μ's effect on the

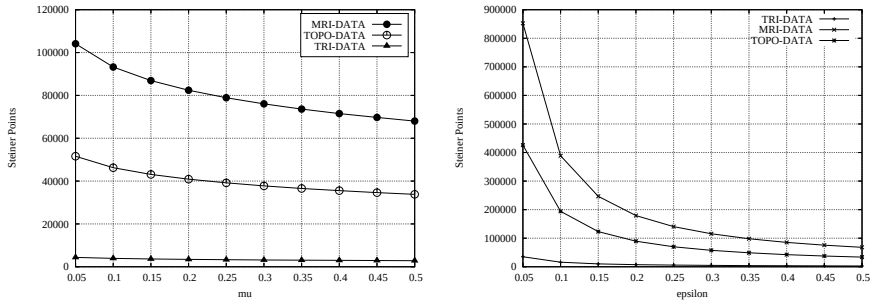


Fig. 8. (a) The effect of fixing ϵ and varying μ. (b) The effect of fixing μ and varying ϵ.

discretization process used in K-Link-Mu. See Fig. 8 (a). Very small values for μ can have a significant impact on Steiner point generation.

In our second series of tests, we fix μ to 0.5 and then vary ϵ to gauge the effect that ϵ has on Steiner point generation. See Fig. 8 (b). Note that once μ is fixed the resulting vertex-vicinity radii for the K-Link and K-Link-Mu algorithms differ by only a constant factor. Although we would like to choose a very small ϵ to guarantee the quality of the k-link path we generate, the number of Steiner points grows quickly as ϵ becomes small.

4.2 Link Generation Performance

In the link generation stage $O((n\delta)^2)$ links must be calculated. In our experiments we computed "exact" links using the prune-and-search scheme (see [9]), which returns an optimal link accurate to any user specified precision.

In the link generation stage we compare the running time for all of the approaches mentioned earlier. See Fig. 9. The "exact" solutions are clearly the most expensive while the heuristic solutions are the least expensive. Despite the fact that the heuristic solutions do not offer any quality guarantees, in our experiments they provided paths very close to those generated by the algorithms that rely on computing approximate links. This is an interesting finding given the expensive nature of link generation.

4.3 Path Finding Performance and Quality

Link generation is the true performance bottleneck in finding k-link paths. But if we assume that discretization and link generation are preprocessing steps to potentially many path finding operations, the performance of the path finding step becomes of critical importance.

In our algorithms the path finding step is straightforward and requires simple, $O(k(n\delta)^2)$ computation.

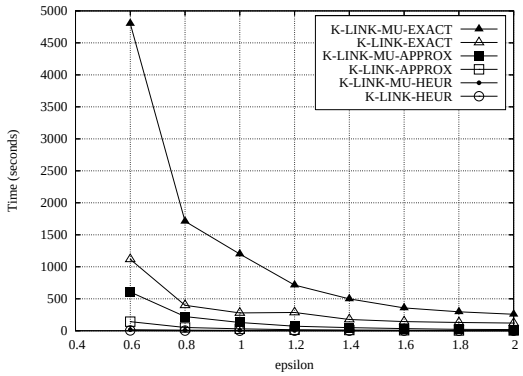


Fig. 9. Timed tests of the link generation stage.

An important issue is the quality of the paths being generated. How closely do generated paths follow the optimal path? How closely do generated paths come to the true or optimal k-link path value in practice?

From our experiments we have seen that when ϵ is quite large there is a tendency for the path to change quite a bit as ϵ changes. Making a relatively small change in ϵ can change the path drastically. However, as ϵ becomes smaller there is a tendency for the path to "lock in" and not change drastically. This also seems to have the effect that tightening ϵ to extremely small values often does not yield proportionally better k-link path lengths.

Fig. 10 is one example of the effect of changing ϵ. Notice how little the path changes between Fig. 10 (c) and (d).

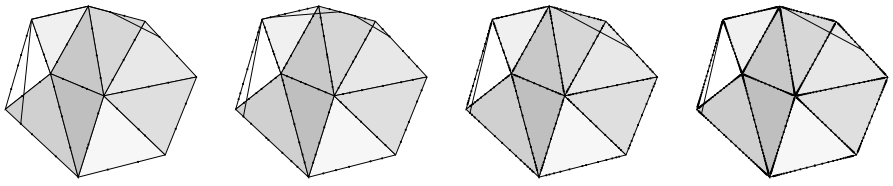


Fig. 10. Four k-link path approximations where $k = 3$ and (a) $\epsilon = 2.0$, (b) $\epsilon = 1.0$, (c) $\epsilon = 0.5$, and (d) $\epsilon = 0.25$.

Changing the number of links, k, a path may utilize is usually more profound. Fig. 11 is an example of the effect of changing k with a fixed ϵ. In this example the path for $k = 2$ is actually not a bad approximation of a path with $k = 9$ in that it crosses all of the same faces. This is less likely to be true when considering more complex subdivisions.

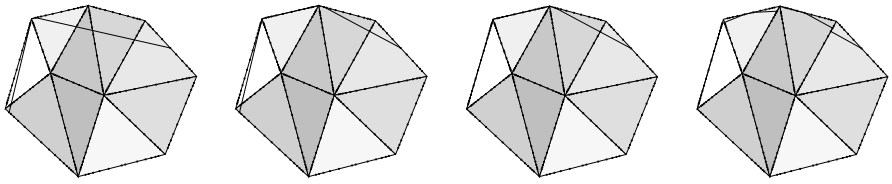


Fig. 11. Four k-link path approximations where $\epsilon = 0.5$ and (a) $k = 2$, (b) $k = 3$, (c) $k = 4$, and (d) $k = 9$.

5 Conclusions

In terms of theory, we believe it will be difficult to improve on the $2k - 1$ link result for an approximating k link path that turns only on edges. Future work that seeks to improve this result may need to consider discretization schemes where Steiner points could also be placed inside the faces of R.

Our experiments highlight many of the difficulties and computation intensive facets of this problem. While it may be difficult to improve on the number of links in an approximating path we believe there is potential for improving the running time for finding solutions using new techniques. For example, while the K-Link-Mu result provides a nicer theoretical result by reducing the number of links in an approximating path, the relationship between μ and Steiner point generation often makes K-Link the more desirable algorithm.

References

1. L. Aleksandrov, M. Lanthier, A. Maheshwari, and J.-R. Sack. An ϵ-approximation algorithm for weighted shortest paths on polyhedral surfaces. In *Proc. 6th Scand. Workshop Algorithm Theory*, vol. 1432 of *Lecture Notes Comput. Sci.*, pages 11–22. Springer-Verlag, 1998.
2. L. Aleksandrov, A. Maheshwari, and J.-R. Sack. Approximation algorithms for geometric shortest path problems. In *Proc. 32nd ACM Sympos. Theory Computing*, pages 286–295, 2000.
3. L. Aleksandrov, A. Maheshwari, and J.-R. Sack. Determining approximate shortest paths on weighted polyhedral surfaces. *Journal of the ACM*, 52(1):25–53, 2005.
4. E. M. Arkin, J. S. B. Mitchell, and C. D. Piatko. Bicriteria shortest path problems in the plane. In *Proc. 3rd Canadian Conf. Computational Geometry*, pages 153–156, 1991.
5. D. Z. Chen, O. Daescu, X. Hu, X. Wu, and J. Xu. Determining an optimal penetration among weighted regions in two and three dimensions. *Journal of Combinatorial Optimization*, 5(1):59–79, 2001.
6. D. Z. Chen, X. Hu, and J. Xu. Optimal beam penetration in two and three dimensions. *Journal of Combinatorial Optimization*, 7(2):111–136, 2003.

7. O. Daescu. Improved optimal weighted links algorithms. In *Proc. ICCS 2nd International Workshop on Computational Geometry and Applications*, pages 65–74, 2002.
8. O. Daescu and J. Luo. Proximity problems on line segments spanned by points. In *Proc. 17th Canadian Conf. Computational Geometry*, pages 224–228, 2005.
9. O. Daescu, J. S. B. Mitchell, S. Ntafos, J. D. Palmer, and C. K. Yap. k-link shortest paths in weighted subdivisions. In *Proc. 9th Workshop on Algorithms and Data Structures*, pages 325–337, 2005.
10. J. Krozel, C. Lee, and J. S. B. Mitchell. Estimating time of arrival in heavy weather conditions. In *Proc. AIAA Guidance, Navigation, and Control*, pages 1481–1495, 1999.
11. M. Lanthier, A. Maheshwari, and J.-R. Sack. Approximating weighted shortest paths on polyhedral surfaces. In *Proc. 13th ACM Sympos. Computational Geometry*, pages 274–283, 1997.
12. M. Lanthier, A. Maheshwari, and J.-R. Sack. Approximating shortest paths on weighted polyhedral surfaces. *Algorithmica*, 30(4):527–562, 2001.
13. C. Mata and J. S. B. Mitchell. A new algorithm for computing shortest paths in weighted planar subdivisions. In *Proc. 13th ACM Sympos. Computational Geometry*, pages 264–273, 1997.
14. J. S. B. Mitchell. Geometric shortest paths and network optimization. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*. Elsevier Science, 2000.
15. J. S. B. Mitchell. Shortest paths and networks. In J. E. Goodman and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry (2nd Edition)*, chapter 27, pages 607–641. Chapman & Hall/CRC, 2004.
16. 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(1):18–73, 1991.
17. J. S. B. Mitchell, C. D. Piatko, and E. M. Arkin. Computing a shortest k-link path in a polygon. In *Proc. 33rd IEEE Sympos. Foundations Computer Science*, pages 573–582, 1992.
18. National Library of Medicine. The visible human project. `http://www.nlm.nih.gov/research/visible`.
19. C. D. Piatko. *Geometric Bicriteria Optimal Path Problems*. Ph.D. thesis, Computer Science, Cornell University, 1993.
20. J. R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In M. C. Lin and D. Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, vol. 1148 of *Lecture Notes Comput. Sci.*, pages 203–222. Springer-Verlag, 1996.
21. Z. Sun and J. H. Reif. Adaptive and compact discretization for weighted region optimal path finding. In *Proc. 14th Sympos. Fundamentals of Computation Theory*, vol. 2751 of *Lecture Notes Comput. Sci.*, pages 258–270. Springer-Verlag, 2003.