Feed-links for Network Extensions[*]

B. Aronov
Dep. of Comp. and Inform. Science
Polytechnic Institute of NYU
Brooklyn, New York, USA
aronov@poly.edu

K. Buchin M. Buchin B. Jansen
Dep. of Inform. and Comp. Sciences
Utrecht University
Utrecht, The Netherlands
[buchin|maike|bmpjanse]@cs.uu.nl

T. de Jong
Dep. of Urban Geography
Utrecht University
Utrecht, The Netherlands
t.dejong@geo.uu.nl

M. van Kreveld M. Löffler J. Luo
Dep. of Inform. and Comp. Sciences
Utrecht University
Utrecht, The Netherlands
[marc|loffler|ljroger]@cs.uu.nl

R. I. Silveira
Dep. of Inform. and Comp. Sciences
Utrecht University
Utrecht, The Netherlands
rodrigo@cs.uu.nl

B. Speckmann
Dep. of Math. and Comp. Science
TU Eindhoven
Eindhoven, The Netherlands
speckman@win.tue.nl

ABSTRACT

Road network data is often incomplete, making it hard to perform network analysis. This paper discusses the problem of extending partial road networks with reasonable links, using the concept of *dilation* (also known as *crow flight conversion coefficient*). To this end, we study how to connect a point (relevant location) inside a polygon (face of the known part of the road network) to the boundary so that the dilation from that point to any point on the boundary is not too large. We provide algorithms and heuristics, and give a computational and experimental analysis.

Categories and Subject Descriptors: H.4 [Information Systems Applications]: *Geographic Information Systems*

General Terms: Algorithms, Experimentation.

Keywords: Road Network, Network Analysis.

1. INTRODUCTION

Motivation. When new schools are built in rural areas of developing countries, an important consideration for their placement is the distance school children have to walk to reach a school – every child must be able to reach some school within a reasonable time. Such spatial analysis is often impossible due to lack of suitable data. Sometimes the locations of settlements are known, but the road network data does not contain any road that leads to the settlement. An example is shown in Figure 1. In reality, however, there will be some path between the settlement and the road network, because people have to reach the (major) roads somehow. In fact, if the land surface permits, there will probably be more than one such path from a settlement, since people generally are unwilling to make large detours.

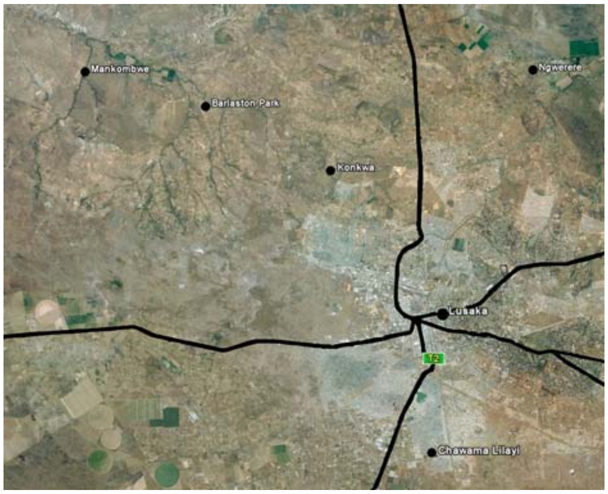

Figure 1: Villages around Lusaka, Zambia, appear disconnected from the road network of Google Earth because smaller roads are missing in the data.

[*] B. Aronov is partially supported by a grant from the U.S.-Israel Binational Science Foundation and by NSA MSP Grant H98230-06-1-0016. K. Buchin, M. Buchin, M. van Kreveld, and J. Luo are supported by the Netherlands Organisation for Scientific Research (NWO) under BRICKS/FOCUS grant no. 642.065.503. M. Löffler and R. I. Silveira are supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 612.065.415. B. Speckmann is supported by the Netherlands Organisation for Scientific Research (NWO) under project no. 639.022.707.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ACM GIS '08, November 5-7, 2008. Irvine, CA, USA
Copyright 2008 ACM ISBN 978-1-60558-323-5/08/11 ...$5.00.

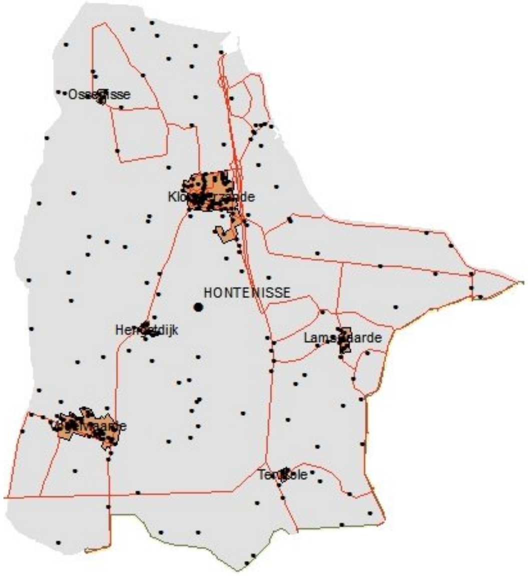

Figure 2: Road network of the Dutch municipality of Hontenisse, showing the centers of the postal code areas (black dots), most of them missed by the roads.

Another scenario where we wish to perform network analysis on a road network and separate locations arises in developed countries. Centers of postal code areas or of municipalities are often used as single point locations for larger groups of houses that lie within a particular postal code or municipality. Computations of travel times between postal code areas or municipalities are not uncommon, since in many situations exact address information is not available. But the centers need not lie on the road network, and therefore must be connected to be able to perform network analysis. A concrete example is shown in Figure 2. The road network of the Dutch municipality of Hontenisse does not suffice to cover the centers of the postal code areas that belong to the municipality.

Even when road network data exists, it may not be available in full detail for many different reasons. National or provincial authorities are often responsible only for roads of a certain level and do not record local or smaller roads in their databases.

All forms of network analysis require a connected network. This applies to measures of centrality of locations, measures of circuitry of the network (the alpha-index), the potential of any destination, and facility location. In most cases, the central concept is that of shortest paths or travel times in the network between any two locations. If a road network is not fully known, and isolated locations exist, a reasonable guess of a possible network that extends the known portion of the network and connects a number of locations is needed. Such a network extension must take the land cover into consideration: Obstacles like lakes or mountains influence the presence and shape of (shortest) paths. Fortunately, data for obstacles is often available.

One can think of a number of practical ways to extend a network to connect all isolated locations. The simplest approach physically snaps the locations to the network. This may produce very unrealistic results if two nearby locations, inside the same face of the network, are snapped to opposite sides of that face. Moreover, this approach modifies the actual positions of the locations, which we assumed to be correct. A different strategy places links from all locations inside a face to a central location, a so-called *feednode*, which is connected directly to the network. The feednode can be the centroid of the face or some other relevant point. Another approach simply connects each individual location to the nearest network node or segment. De Jong and Tillema [5] use a Delaunay network to connect all locations. They discard Delaunay edges that cross obstacles and merge the remaining Delaunay network with the road network. None of these approaches tries to control the amount of detour that the resulting extended network can have, and therefore, estimates based on such extensions can be very unreliable.

Detours. A measure of the detour between two points forced by a network is the *dilation* or *crow flight conversion coefficient* (CFCC). It is defined as the quotient between the network distance and the Euclidean distance. Previous studies in developed countries have shown that the CFCC of road networks usually varies between 1.13 and 1.45 [13]. Networks in developing countries are expected to have a higher coefficient, although one could claim, for instance, that between any two locations at some distance, a route on paths exists that typically has at most twice the length of the Euclidean or crow flight distance.

If a network is such that for any two points p, q on the network, their shortest path is at most t times as long as the Euclidean distance between p and q, then the network has a dilation, or crow flight conversion coefficient, of t (and t is the smallest number for which this holds). For example, the graph of a 3×3 regular square grid has a dilation of 2, because for any two points on the grid lines, a shortest path is at most twice as long as the Euclidean distance. The dilation of 2 is realized by two middles of opposite sides of a square cell. An equilateral triangle also has a dilation of 2, while a circle has a dilation of $\pi/2 \approx 1.57$, see Figure 3.

Dilation is a concept that has been studied extensively in computational geometry [2, 3, 6, 7, 10, 12, 14]. The basic problem is to compute – for a given set of points – a graph in which a shortest path between any two points is close to their Euclidean distance. Such a graph is called a t-spanner if it has dilation t. For a detailed study on spanners see the book by Narasimhan and Smid [11]. Spanners in the presence of obstacles have also been considered [1, 4]. Recently, Farshi and Gudmundsson [8] performed an extensive experimental comparison between various available spanner algorithms.

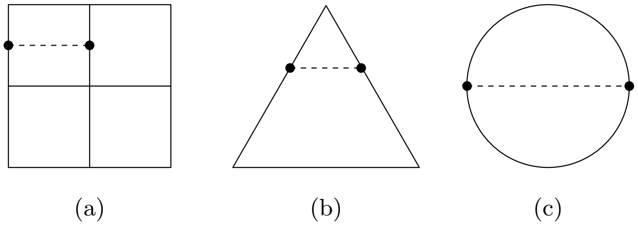

Figure 3: Dilation in (a) a square, (b) an equilateral triangle, and (c) a circle. The dashed lines indicate pairs of points achieving the dilation.

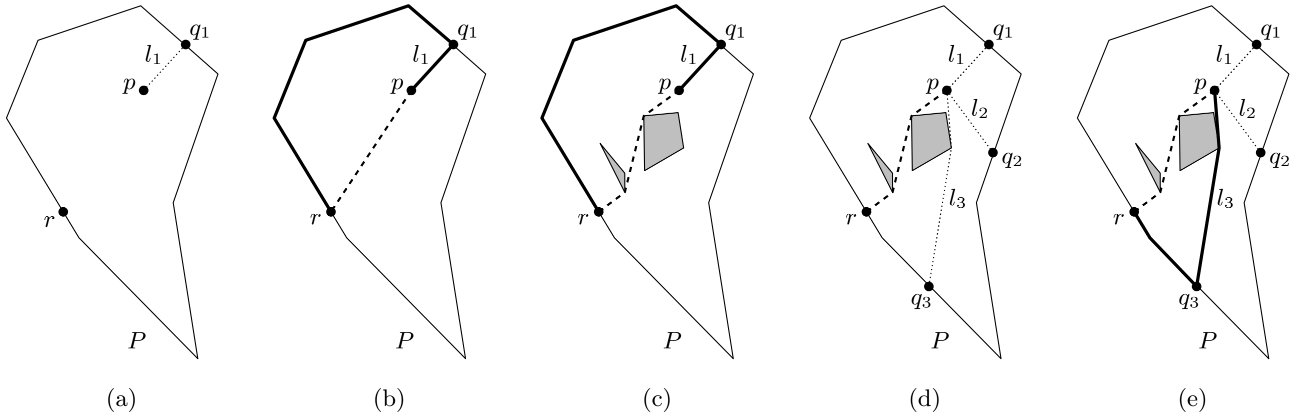

Figure 4: (a) Example of a setting with one feed-link l_1 (dotted). (b) The dilation of point r is the length of the path between p and r using l_1 and part of the boundary of P (solid thick), divided by the length of the shortest path between them (dashed). (c) Obstacles must be taken into account for computing shortest paths. (d) Example of a setting with three feed-links. (e) The feed-link that gives the shortest path is used in the computation of the dilation, in this case l_3.

Results. In this paper we study the problem of connecting a single point p to a road network by a *feed-link*: A connection between p and a point on the road network that avoids obstacles (lakes, swamps, mountains, etc.). We assume that p lies in a simple polygon P, which is a face of the known road network. Inside P there may be obstacles which are represented by simple polygons.

We show how to compute one or more feed-links from point p to the boundary of P using various algorithms and analyze the resulting dilation for p. Since p is our point of interest, we concentrate on the dilation from p only. We consider both the dilation from p to any point on the boundary of P and the dilation from p to a fixed set of points (for example, the vertices) of P.

In particular, in Section 2 we explore the effect one or more feed-links can have on the dilation from a theoretical point of view. In Section 3 we give efficient algorithms that compute the dilation for a set of k feed-links with or without obstacles. We also describe several heuristics that place one or more feed-links to minimize dilation. In Section 4 we compare these heuristics experimentally on a large set of polygons with obstacles. Our experiments show that a greedy approach works very well. In Section 5 we consider the problem of finding a minimum size set of feed-links that achieves a given dilation. We give an additive 1-approximation algorithm that solves this problem with or without obstacles. Finally, in Section 6 we discuss various extensions to our algorithms. We show how to combine the solutions for single faces into a solution for a complete road network, we consider the case of several isolated locations inside the same face, and we adapt our algorithm to non-simple polygons representing areas with dead-end roads or cul-de-sacs.

2. FEED-LINKS AND DILATION

Our input is a polygon P, a collection $\mathcal{B} = \{B_1, \ldots, B_h\}$ of obstacles that are disjoint and do not cross the boundary of P, and a point p that lies inside P but outside all B_i. Our goal is to find a feed-link that connects p to some point on the boundary of P while avoiding the obstacles in $\mathcal{B}$.

We first need to redefine the concept of dilation in the presence of obstacles. Let r and q be points on the boundary of P. We denote by sp-length(r, p) the length of a shortest path between r and p that avoids the obstacles. Further we denote by bd-length(r, q) the distance between r and q on the boundary of P, i.e., the length of the shorter of the two paths between r and q along the boundary of P.

Let $q_1, \ldots, q_k$ be a set of k points on the boundary of P that have feed-links $l_1, \ldots, l_k$ to p. The *dilation $\delta(r)$ of the point* r is the length of the shortest path from r to p using only the boundary of P and one of the feed-links from $l_1, \ldots, l_k$, divided by sp-length(r, p), i.e.,

$$\delta(r) := \frac{\min_{i=1,\ldots,k}(\text{sp-length}(p, q_i) + \text{bd-length}(r, q_i))}{\text{sp-length}(p, r)} .$$

The *dilation obtained by the set of feed-links* is the maximum dilation realized by any point r on the boundary of P.

Figure 4 shows a polygon P with a single feed-link (a), the resulting dilation visualized for a point r on the boundary (b), the resulting dilation visualized if P contains obstacles (c), a situation with three feed-links (d), and the dilation of r in that case (e).

To achieve a small dilation, a feed-link should be chosen as a shortest path from some point on the boundary of P to p. Obviously, this shortest path must avoid the obstacles. If P is not convex, then the shortest path from a boundary point q to p might pass through the outside of P, and hence we might have to avoid obstacles outside P as well. If all feed-links are indeed shortest paths, then the points where a feed-link is attached have dilation exactly 1, whereas all other points have dilation at least 1.

We first assume that only one feed-link is present from p to the boundary of P, namely to the closest point on P. This seems a reasonable choice and in fact we can show:

LEMMA 1. *If p has one feed-link to the closest point on the boundary of P, then the resulting dilation obtained by the feed-link is never worse than twice the dilation obtained by an optimally placed feed-link.*

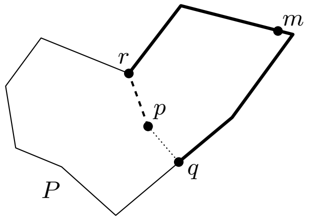

Figure 5: Illustration of proof of Lemma 1: connecting p to its closest point results in a maximum dilation at most twice the optimal one.

PROOF. Suppose that the closest point is q, and that a point r has the worst dilation when a feed-link between q and p is chosen. Since q is the only feed-link the dilation $\delta(r)$ of r is

$$\delta(r) = \frac{\text{sp-length}(p,q) + \text{bd-length}(r,q)}{\text{sp-length}(r,p)} .$$

We need to prove that for any feed-link there is a point with dilation at least $\delta(r)/2$. For this we consider several cases depending on which part of the boundary a feed-link connects to. Let m be the point in the middle of the shorter boundary path from r to q (see Figure 5). If a feed-link connects to a point between q and m, then the length of the detour to r is at most halved and therefore the dilation of r is at least $\delta(r)/2$. This is in particular the case for a feed-link connecting to m. In this case the dilation of q is at least the dilation of r, and therefore at least $\delta(r)/2$, since the length of the detour is the same and q is a closest point to p. Now if a feed-link connects to a point between m and r, then the dilation of q is at least as large as for a feed-link to m and therefore at least $\delta(r)/2$. Finally, if a feed-link connects to a point on the longer boundary part between q and r, then the same arguments apply. □

The bound in Lemma 1 is tight in the sense that the factor by which the dilation using the closest point is worse than the dilation for the optimal feed-link can be arbitrarily close to 2. This is illustrated in Figure 6: q and r are the closest points to p and taking a feed-link to one of them gives a dilation of $4x + 7$ at the other. The optimal feed-link is between p and q' and gives a dilation of $2x + 5$ (obtained at q and r). Thus, for $x \to \infty$ the factor approaches 2.

We can relate the dilation of a point r in a polygon P with obstacles to the dilation of r in P without obstacles.

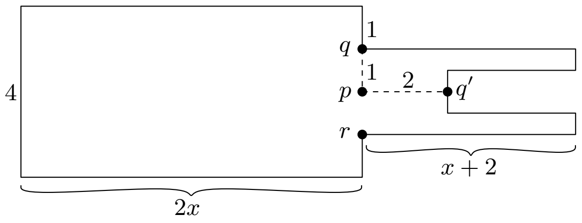

Figure 6: Example where choosing the closest point (q) to connect the feed-link results in a maximum dilation close to two times worse than when connecting to the optimum feed-link point (q').

LEMMA 2. *Let $\delta(r)$ be the dilation of r with feed-link l for a polygon P without obstacles, and let $\delta'(r)$ be the dilation of r for the same situation, but now P may contain obstacles and l avoids the obstacles. Then $\delta'(r) \leq 3 \cdot \delta(r)$.*

PROOF. Let q be the point on the boundary of P where l connects to. The dilation $\delta(r)$ is given by

$$\delta(r) \;=\; \frac{\text{dist}(p,q) + \text{bd-length}(r,q)}{\text{dist}(p,r)} ,$$

and the dilation $\delta'(r)$ is given by

$$\delta'(r) \;=\; \frac{\text{sp-length}(p,q) + \text{bd-length}(r,q)}{\text{sp-length}(p,r)} .$$

Using the fact that

$$\text{sp-length}(p,q) \leq \text{sp-length}(p,r) + \text{bd-length}(r,q)$$

we get

$$\begin{aligned}
\delta'(r) &= \frac{\text{sp-length}(p,q) + \text{bd-length}(r,q)}{\text{sp-length}(p,r)} \\
&\leq \frac{\text{sp-length}(p,r) + 2 \cdot \text{bd-length}(r,q)}{\text{sp-length}(p,r)} \\
&= 1 + \frac{2 \cdot \text{bd-length}(r,q)}{\text{sp-length}(p,r)} \\
&\leq 1 + \frac{2 \cdot \text{bd-length}(r,q)}{\text{dist}(p,r)} \\
&= \frac{\text{dist}(p,r) + 2 \cdot \text{bd-length}(r,q)}{\text{dist}(p,r)} .
\end{aligned}$$

Since $\text{dist}(p,r) \leq \text{dist}(p,q) + \text{bd-length}(r,q)$ we get

$$\begin{aligned}
\delta'(r) &\leq \frac{\text{dist}(p,q) + 3 \cdot \text{bd-length}(r,q)}{\text{dist}(p,r)} \\
&\leq 3 \cdot \frac{\text{dist}(p,q) + \text{bd-length}(r,q)}{\text{dist}(p,r)} \;=\; 3 \cdot \delta(r) .
\end{aligned}$$

This completes the proof. □

If we want to guarantee that a network has a dilation that is bounded from above by a fixed value, we may have to use several feed-links to achieve this. In such a case, we would like to use the minimum number k of feed-links (in Section 5 we address the algorithmic problem). A dilation of 2, for instance, is often a reasonable assumption in a realistic road network. It is interesting to know how many feed-links need to be placed in the worst-case, i.e., how large k needs to be for a given dilation. In particular, we may want to know

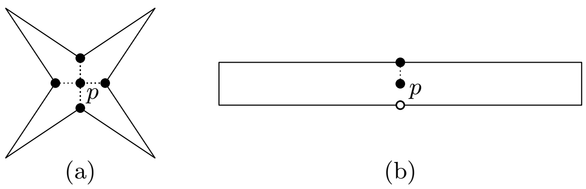

Figure 7: Dilation in a (a) non-convex polygon and (b) convex polygon. The dotted lines inside the polygons show feed-links and the circle in (b) indicates a point with large dilation.

whether we can bound k by a constant or not. It is easy to see that simple polygons exist that require many feed-links to bound the dilation. Consider a polygon that is shaped like a star with $n/2$ spikes, as in Figure 7 (a). If the spikes are sufficiently long, we need $n/2$ feed-links to bound the dilation of the polygon by a given constant c, and hence, a constant number of feed-links does not always suffice.

For convex polygons, the dilation for one feed-link also cannot be bounded by a constant. Consider a skinny rectangle with p the midpoint of the rectangle, as in Figure 7 (b). If the one feed-link connects to a point above p then the dilation for a point below p is larger than any given constant c if the rectangle is sufficiently wide. The analogue holds for the case where the one feed-link connects to a point below p. Thus, if we place only one feed-link, we may get an arbitrarily large dilation. However, we can prove that for any convex polygon there is a placement of two feed-links such that the dilation is bounded by 6. More generally, we can show that there is a placement of k feed-links such that the worst dilation is $1 + O(\frac{1}{k})$ (proof omitted here), but this is mainly of theoretical interest.

3. ALGORITHMS FOR FEED-LINKS

In this section we present algorithms that compute the dilation and algorithms that place feed-links.

3.1 Algorithms for dilation and shortest paths

First we show how to compute the dilation for a given set of feed-links $l_1, \ldots, l_k$ efficiently in case there are no obstacles.

Assume that the points $q_1, \ldots, q_k$ where the feed-links attach to the boundary of P are sorted along the boundary. For any two consecutive points q_i and q_{i+1}, find the point m_i on the boundary of P where the network distance to p via feed-link l_i is equal to the network distance via feed-link l_{i+1}. See Figure 8. Then along the boundary of P, we have points $q_1, m_1, q_2, m_2, \ldots, q_k, m_k$. All points between m_j and m_{j+1} will have their best network connection to p via q_{j+1}.

For any point on an edge of the boundary of P between m_j and m_{j+1}, the network distance changes linearly in the position of that point on the edge, and the Euclidean distance changes hyperbolically. Therefore, an analytic computation can determine the location on the edge where the dilation is maximized: If we parameterize the edge by $t \in [0, 1]$, then the network distance is a linear function $at + b$ where $a > 0$ and $b > 0$ are reals depending only on P, p, and q_{j+1}. The Euclidean distance has the form $\sqrt{At^2 + Bt + C}$ where A, B, and C are constants depending only on the coordinates of p and the endpoints of the edge. By setting the derivative of the quotient to zero, we get a parameter value $t = (-bB + 2aC)/(2Ab - aB)$ of a possible maximum. If $t \in [0, 1]$ then we insert it into the quotient to determine the dilation at the corresponding point on the edge. Otherwise we check the dilation for $t = 0$ and $t = 1$. Finally we compare the result to the largest dilation found so far.

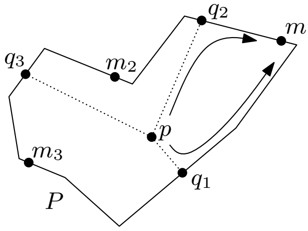

Figure 8: At the points m_1, m_2, and m_3, the used feed-link changes.

The computation of the maximum is done for all edges between m_j and m_{j+1}, and similarly, for all pairs of consecutive midpoints of this type.

If P has n edges, then splitting at the points $q_1, \ldots, q_k$ and $m_1, \ldots, m_k$ gives rise to at most $n + 2k$ edges on which we maximize the dilation, taking constant time for each. Therefore, we can compute the dilation of the polygon and its feed-links in $O(n + k)$ time. If we need to compute the sorted order of $q_1, \ldots, q_k$ on the boundary, we must add an $O(k \log k)$ term. However, note that k is typically a small constant, and the dependency on n is the relevant part.

THEOREM 1. *Given a simple polygon P with n vertices and a set of k feed-links, we can compute the dilation in $O(n + k \log k)$ time.*

The situation is more difficult if there are obstacles in P. We can use the algorithm of Hershberger and Suri [9] to find shortest paths amidst obstacles in $O(n \log n)$ time. Since we are interested only in shortest paths to p, we run this algorithm on p and the obstacles $B_1, \ldots, B_h$. The algorithm will compute a subdivision of the plane into cells where the first vertex of an obstacle on the shortest path to p is fixed. This subdivision S has linear complexity (linear in the number of vertices b of all obstacles) and can be computed in $O(b \log b)$ time.

We overlay the subdivision S with the polygon P to partition P's edges into subedges that have a similar shortest path to p, similar in the sense that the first obstacle vertex on the shortest path to p is the same (see Figure 9). This allows us to get an analytic expression for the length of the shortest path from any point on the boundary of P to p of the form $\sqrt{At^2 + Bt + C} + D$ for constant reals A, B, C, and D. We can now find the dilation of the set of feed-links by applying the ideas from the case where no obstacles were present. This analytical expression gives rise to two candidate solutions for the maximum dilation, in contrast to the case of no obstacles where there is only one candidate.

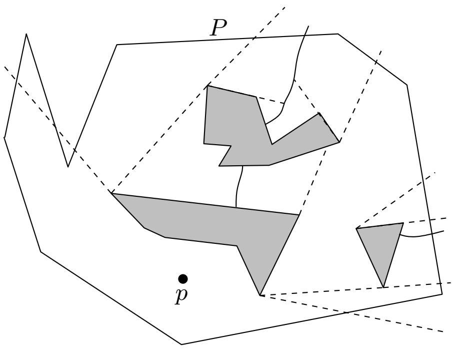

Figure 9: Example of the overlay of subdivision S and the polygon P.

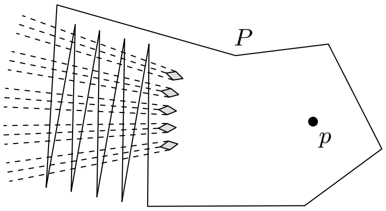

Figure 10: Theoretically, the overlay of S and P can have $\Theta(nb)$ complexity. In practice, this is unlikely.

Theorem 2. *Given a simple polygon P with n vertices, a set of obstacles with b vertices, and a set of k feed-links with k' edges in total, we can compute the dilation in $O(nb + k' + k \log k + b \log b)$ time.*

Proof. Most terms in the time bound are clear from the steps needed in the algorithm. The quadratic term is caused by the overlay of S and the boundary of P: There may be $O(nb)$ intersection points in the overlay, and therefore the edges of P are partitioned into $O(nb)$ pieces due to the shortest path subdivision. For each piece, we can find the point with maximum dilation on that piece in constant time. □

The algorithm above also finds the closest point on the boundary of P to the point p in $O(nb)$ time in a polygon with obstacles, which we can use to compute a good feed-link. Without obstacles, this operation can easily be done in $O(n)$ time. We also note that even for one obstacle with b vertices, the overlay of S and P can have $\Theta(nb)$ complexity in theory. However, to obtain this complexity, the polygon and the obstacle have to be laid out in very contrived configurations, similar to the one depicted in Figure 10. In practice we expect the complexity of the overlay to be much lower.

3.2 Heuristics for placing feed-links

We next present two heuristics for placing one or more feed-links to achieve a small dilation.

The first heuristic is called the *greedy heuristic*. If only one feed-link is required, we connect p to the closest point on the boundary of P. If more than one feed-link is required, we first add a feed-link to the closest point, then iteratively find the point with worst dilation, and add a feed-link to this point.

The second heuristic is called the *sector heuristic*. Assume we wish to place k feed-links. Then k half-lines originating at p are chosen, all with angle $2\pi/k$ in between. This partitions the plane into k wedges with equal opening angle. In every wedge, we determine the point on the boundary of P that is closest to p, i.e., has the shortest path to p avoiding obstacles, and place a feed-link to that point.

Theorem 3. *For a polygon P without obstacles, the greedy heuristic and the sector heuristic take $O(n)$ time to find $k = O(1)$ feed-links, where n is the number of vertices of P. For a polygon P with obstacles, the greedy heuristic and the sector heuristic run in $O(nb + b \log b)$ time, where b is the number of vertices of the obstacles.*

The sector approach does not specify a base angle for the k half-lines, that is, they can be rotated around p as a whole and still give wedges with equal opening angles. We may just choose a base angle at random. We may also place the half-lines after computing the closest point, and choose the base angle such that the closest point is in the middle of a sector. We call these two methods the random and the positioned sector heuristics, respectively. The positioned heuristic makes sure, for instance, that when placing only two feed-links to points q_1 and q_2, the angle $\measuredangle q_1 p q_2 \geq \pi/k = \pi/2$, whereas for the random heuristic this angle can be arbitrarily small, and q_1 and q_2 may be very close.

We note that it is unlikely in practice that the heuristics require $\Theta(nb)$ time for a polygon with obstacles. The overlay of the shortest path map and the polygon P can be expected to have much fewer vertices than $\Theta(nb)$, and these vertices can be computed in an output-sensitive manner (for example by a plane sweep algorithm).

4. EXPERIMENTAL RESULTS

We have implemented the three heuristics (greedy and the two sector heuristics) described in the previous section, and have run tests on a collection of 100 polygons with two obstacles. The objective is to determine the dilation of typical polygons one can expect in practice when using one, two, or more feed-links, and to compare which feed-link placement heuristic gives the best resulting dilation.

First we implemented a polygon and obstacles generator and adjusted it to generate a collection that (visually) seems reasonable for the application at hand: Incomplete road networks. Also for this reason, we did not consider more than two obstacles in the polygon. Figure 11 shows various polygons that were generated. Then we chose a random point p in each polygon, and ran the heuristics for $1, \ldots, 10$ feed-links. Table 1 shows the results.

For one feed-link, all three heuristics will choose the same feed-link, so the results are the same. For more feed-links, it appears that the greedy heuristic outperforms the other two. Figure 12 shows four examples of the greedy heuristic, run on the same polygon for different numbers of feed-links. The two sector heuristics perform comparable with

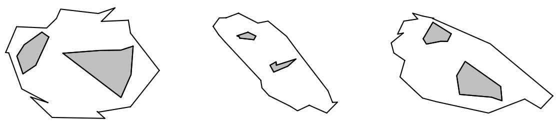

Figure 11: Randomly generated polygons with two obstacles.

k	random sector		posit. sector		greedy	
	μ	σ	μ	σ	μ	σ
1	4.183	1.860	4.183	1.860	4.183	1.860
2	2.747	1.122	2.176	0.534	2.086	0.382
3	2.129	0.651	2.288	0.759	1.815	0.261
4	1.897	0.351	1.826	0.272	1.649	0.149
5	1.729	0.230	1.680	0.217	1.553	0.122
6	1.622	0.166	1.612	0.180	1.493	0.091
7	1.579	0.165	1.564	0.167	1.447	0.063
8	1.531	0.155	1.538	0.160	1.419	0.055
9	1.495	0.137	1.486	0.163	1.392	0.041
10	1.473	0.153	1.447	0.155	1.380	0.046

Table 1: The mean μ and standard deviation σ of the dilation values for k feed-links.

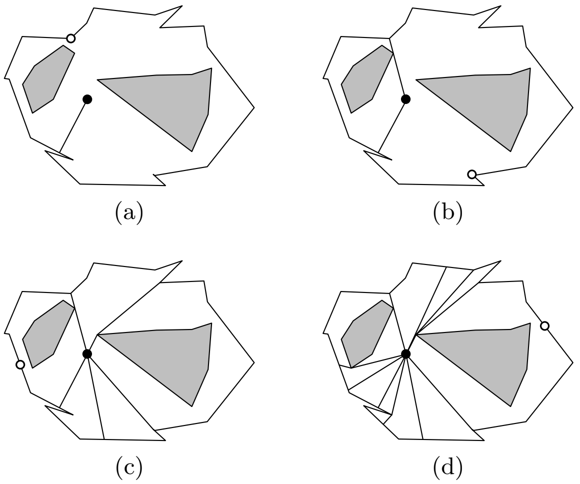

Figure 12: Four examples of the greedy heuristic with different numbers of feed-links. The circle indicates the point with the worst dilation. (a) One feed-link, dilation is 4.0513. (b) Two feed-links, dilation is 2.5581. (c) Five feed-links, dilation is 1.7753. (d) Ten feed-links, dilation is 1.3137.

respect to each other, although the positioned sector heuristic seems to work better for two feed-links and the random sector heuristic seems to work better for three feed-links. The (average) dilation decreases with more feed-links, which is to be expected. Already for three feed-links, the dilation obtained by the greedy heuristic is below 2 on the average.

Table 1 shows how the three heuristics compare to each other, but it does not show how close they get to the best achievable dilation. To determine this, we have implemented more advanced methods that are computationally more demanding. Using observations to obtain lower bounds on the boundary length needed, the advanced methods yield both upper and lower bounds on the smallest dilation that can be attained with any set of k feed-links for a given polygon P and point p in P. However, due to the computational cost, we only have such bounds for $k = 1, \ldots, 4$. They are given in Table 2. We observe that the dilation bounds are not as tight for more feed-links. We also observe that the greedy heuristic is about $10\% - 20\%$ off the optimum dilation, for $k = 1, \ldots, 4$. For more feed-links, we do not have data.

k	lower bound μ_{opt}	upper bound μ_{opt}
1	3.7744	3.7753
2	1.8758	1.8776
3	1.5710	1.5956
4	1.3862	1.4854

Table 2: Bounds on the average optimal dilation.

5. MINIMIZING THE NUMBER OF FEED-LINKS FOR A GIVEN DILATION

In this section we study the following problem: Given a polygon P, a point p inside P, and a target dilation $c > 1$, find a (small) set of feed-links that connect p to the boundary of P to achieve a dilation of at most c.

An exact algorithm for this problem seems hard to obtain; however, we give a simple algorithm that finds a set of feed-links that contains at most one feed-link more than an optimal set. The algorithm proceeds in a greedy fashion. We start by choosing an arbitrary first link pq_1. For instance, we can choose the closest point q_1 to p on the boundary of P. We want to place the next feed-link q_2 (in clockwise order) as far away from q_1 as possible such that all points between q_1 and q_2 have a dilation (via q_1 or q_2) not larger than c.

For this we first traverse the boundary of P starting at q_1 until we reach a point m_1 for which another point just beyond m_1 has a dilation larger than c via q_1, as in Figure 8. To find the point m_1 we traverse the edges of P, starting at q_1. As in Section 3.1 we check for a maximum of the dilation on the edge, and whether it is larger than c. If it is, we find the point m_1 where dilation reaches c by solving the corresponding quadratic equation with the edge parameterized by $t \in [0, 1]$.

Then we continue traversing the boundary of P to find the point q_2. The point q_2 is maximal (in the sense of furthest from m_1) with the property that all points in between q_2 and m_1 have dilation not larger than c via q_2. Let ℓ be the network distance from m_1 to p via q_1. Then the furthest possible placement for q_2 is such that the network distance from m_1 to p via q_2 equals ℓ, as well. However, we might have to place q_2 closer to q_1 than this, since it is possible that some point between m_1 and q_2 still has a higher dilation. We check this by hypothetically placing q_2 there (i.e., we do not actually compute its exact placement, we just remember that the distance to p via this link is ℓ from m_1) and continue traversing the edges starting at m_1. For each edge, we check whether the dilation on this edge via (the current hypothetical) q_2 is not larger than c. If this check fails, i.e., the dilation exceeds c on the current edge, we re-compute the (hypothetical) q_2 (which will then be closer than before).

We also check whether we have not passed q_2 yet (since we did not actually compute its placement). For this we check whether the boundary length of m_1 to p via a feed-link on the current edge of the traversal is ℓ. If we pass the hypothetical q_2, we place a feed-link, and continue in the same way to place the next feed-link. This process continues until we have traversed the whole boundary of P.

THEOREM 4. *The above algorithm is an additive* 1*-approximation for computing the minimum number of feed-links for a given dilation. It runs in $O(n+k)$ time for polygons without obstacles and in $O(nb+k+b\log b)$ time for polygons with obstacles, where n is the number of vertices of the polygon, the set of obstacles has b vertices, and k is the minimum number of feed-links required.*

PROOF. Assume the algorithm places $k+1$ feed-links at $q_1, \ldots, q_{k+1}$ ($k \geq 0$). By definition of the q_i, any set of feed-links achieving a dilation at most c needs to have a feed-link in each of the k sectors between q_i and q_{i+1} for $i = 1, \ldots, k$.

For polygons without obstacles the running time of the algorithm is $O(n+k)$ since it spends constant time per edge of the polygon with an additional overhead of at most $k+1$ for placing the feed-links. For polygons with obstacles we use the same approach as described in Section 3.1. That is, we overlay the polygon P with the subdivision S which has $O(nb)$ complexity. We then run the algorithm as before on the $O(nb)$ edges. □

6. EXTENSIONS

The problem of determining feed-links for a single point inside a polygon is considerably more restricted than the problem that motivated us to study feed-links. Generally we have a road network and a number of locations inside its faces, and wish to determine a reasonable set of connections, feed-links, from the locations to the network. The general problem extends our simple version in at least two ways. Firstly, the road network has more than one face to consider, and secondly, there may be more than one unconnected location inside a face of the road network.

To deal with multiple faces that contain unconnected locations, we can simply treat each face separately. In some cases we can use the dilations of different faces to bound the dilation between two locations in these faces, see Figure 13 (a).

Lemma 3. *Given a simple polygon P_1 with obstacles, a point p_1, and a set of feed-links that results in a dilation of δ_1. Similarly, given P_2, p_2, and a set of feed-links that results in a dilation of δ_2. If P_1 and P_2 are adjacent faces in the road network, and the shortest path between p_1 and p_2 lies fully in $P_1 \cup P_2$, then the dilation for p_1 and p_2 is at most* $\max(\delta_1, \delta_2)$.

Proof. Since the shortest path between p_1 and p_2 lies fully in $P_1 \cup P_2$, it intersects the common boundary of P_1 and P_2. Let r be a point where the shortest path intersects the common boundary. The dilation in P_1 from p_1 to r is bounded by δ_1, and the dilation in P_2 from p_2 to r is bounded by δ_2. Therefore, the dilation between p_1 and p_2 is at most $\max(\delta_1, \delta_2)$. □

The lemma is no longer true if the shortest path intersects other faces of the road network. If we then still wish to bound the dilation, we would also need a bound on the dilation between any two points on the boundary of each face, not just the dilation from the points inside to the boundary, see Figure 13 (b).

When more points lie in a single polygon, we should connect all of them to the boundary or to each other. In this case we are not only interested in the dilation of each point to the boundary, but also between different points in the polygon. In applications we do not expect a large number of points in each polygon.

Since the closest point idea and the greedy heuristic work well for the single-point case, we extend them to multiple points. Let $p_1, \ldots, p_m$ be a set of unconnected points inside a polygon P with obstacles. We repeatedly choose an unconnected point that is closest to the boundary of P or to an existing feed-link (which includes already connected points), and add the connection as a feed-link. After adding as many feed-links as there were unconnected points, we have a connected network in the polygon, and can add more feed-links greedily to reduce the dilation. We compute a pair of points, one from $p_1, \ldots, p_m$ and one from either the boundary of P, or from $p_1, \ldots, p_m$ as well, that gives the worst dilation, and connect them with a shortest path. It is plausible that such an approach works well, but both algorithmic and experimental research is needed to substantiate this.

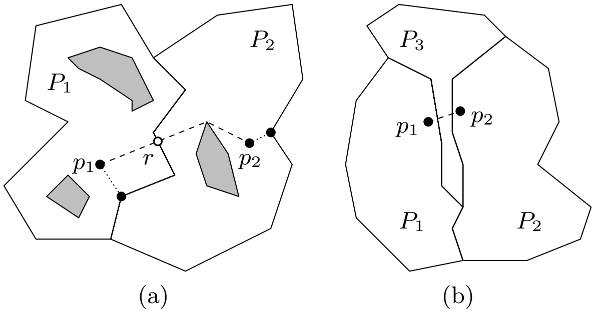

Figure 13: The dilation of two points in different faces.

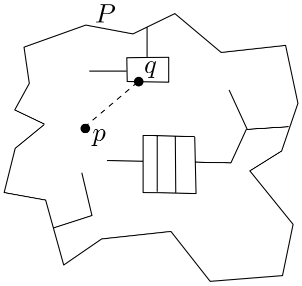

Figure 14: Example of a polygon P with cul-de-sacs and dead-end roads.

Finally, we consider the case that the polygon P is not a simple polygon, for example, P may be part of an area with many cul-de-sacs and dead-end roads (see Figure 14). Our algorithms can handle this situation with only minor modifications.

7. CONCLUSIONS

We studied the problem of extending a partial road network by adding feed-links to disconnected locations. We formalized this problem by optimizing the *dilation*, which – to the best of our knowledge – has not been attempted in previous work. We gave several theoretical results for the specific case when there is just one disconnected location, and implemented three different heuristics that perform well in practice. The heuristics have a quadratic worst case running time, but in typical situations the running time will be near-linear. The methods can be extended to the more general case, where the road network has multiple faces and multiple locations inside each face.

8. REFERENCES

[1] M. A. Abam, M. de Berg, M. Farshi, and J. Gudmundsson. Region-fault tolerant geometric spanners. In *Proc. 18th Symposium on Discrete Algorithms*, pages 1–10, 2007.

[2] P. K. Agarwal, S. Har-Peled, and M. Karia. Computing approximate shortest paths on convex polytopes. *Algorithmica*, 33(2):227–242, 2002.

[3] H.-K. Ahn, M. Farshi, C. Knauer, M. H. M. Smid, and Y. Wang. Dilation-optimal edge deletion in polygonal cycles. In *Proc. 18th International Symposium on Algorithms and Computation*, LNCS 4835, pages 88–99, 2007.

[4] S. R. Arikati, D. Z. Chen, L. P. Chew, G. Das, M. Smid, and C. D. Zaroliagis. Planar spanners and approximate shortest path queries among obstacles in the plane. In *Proc. 4th European Symposium on Algorithms*, LNCS 1136, pages 514–528, 1996.

[5] T. de Jong and T. Tillema. Transport network extensions for accessibility analysis in geographic information systems. In *Proceedings Africa GIS 2005*, 2005.

[6] A. Ebbers-Baumann, A. Grüne, and R. Klein. The geometric dilation of finite point sets. *Algorithmica*, 44(2):137–149, 2006.

[7] M. Farshi, P. Giannopoulos, and J. Gudmundsson. Finding the best shortcut in a geometric network. In *Proc. 21th Symposium on Computational Geometry*, pages 327–335, 2005.

[8] M. Farshi and J. Gudmundsson. Experimental study of geometric t-spanners: A running time comparison. In *Proc. Workshop on Experimental Algorithms*, LNCS 4525, pages 270–284, 2007.

[9] J. Hershberger and S. Suri. An optimal algorithm for Euclidean shortest paths in the plane. *SIAM Journal on Computing*, 28(6):2215–2256, 1999.

[10] S. Langerman, P. Morin, and M. A. Soss. Computing the maximum detour and spanning ratio of planar paths, trees, and cycles. In *Proc. 19th Symposium on Theoretical Aspects of Computer Science*, LNCS 2285, pages 250–261, 2002.

[11] G. Narasimhan and M. Smid. *Geometric Spanner Networks*. Cambridge University Press, 2007.

[12] G. Narasimhan and M. H. M. Smid. Approximating the stretch factor of euclidean graphs. *SIAM Journal on Computing*, 30(3):978–989, 2000.

[13] J. R. Ritsema van Eck. *Analyse van Transportnetwerken in GIS voor Sociaal-Geografisch Onderzoek*. PhD thesis, Utrecht University, 1993.

[14] C. Wulff-Nilsen. Computing the dilation of edge-augmented graphs in metric spaces. In *Abstr. 24th Europeon Workshop on Computational Geometry*, pages 123–126, 2008.