# Computing the Detour of Polygons

Ansgar Grüne Rolf Klein Elmar Langetepe

**Abstract**

Let $P$ be a simple polygon in $\mathrm{R}^2$ with $n$ vertices. The detour of $P$ between two points, $x, y \in P$, is the length of a shortest path contained in $P$ and connecting x to y, divided by the distance of these points. The detour of the whole polygon is the maximum detour between any two points in $P$. We first analyze properties of pairs of points with maximum detour. Next, we use these properties to achieve a deterministic $O(n^2)$-algorithm for computing the maximum Euclidean detour and a deterministic $O(n \log n)$-algorithm which calculates a $(1+\varepsilon)$-approximation. Finally, we consider the special case of monotone rectilinear polygons. Their $L^1$-detour can be computed in time $O(n)$.

## 1 Introduction

Let $P$ be a connected set in $\mathrm{R}^d$. For any two points $x, y \in P$ let $d_P(x, y)$ denote the infimum of the lengths of all curves which are contained in $P$ and connect $x$ to $y$. The length of the curves is measured using a given norm $\|.\|$. The *detour* $\delta_P(x, y)$ between $x$ and $y$ in $P$ with respect to $\|.\|$ and the detour $\delta(P)$ of $P$ are defined as

$$\delta_P(x, y) := \frac{d_P(x, y)}{\|y - x\|}, \qquad \delta(P) := \sup_{x,y \in P, x \neq y} \delta_P(x, y).$$

Narasimhan and Smid [8] examined the problem of computing a value similar to $\delta(G)$ for a given Euclidean graph $G$. They restricted the maximum to pairs of vertices. Thus, their problem is slightly different (but not necessarily easier). The maximum of all points was first considered by Ebbers-Baumann et al. [3]. They presented an $O(n \log n)$ approximation algorithm for $n$-link chains in $\mathrm{E}^2$. Later, Agarwal et al. [1] gave a randomized $O(n \log^3 n)$ and a deterministic $O(n \log^4 n)$ exact algorithm. Simultaneously, Langerman, Morin and Soss [6] constructed an $O(n \log n)$ randomized algorithm for solving the same problem.

In this abstract we present algorithms computing the detour of a simple polygon $P \subset \mathrm{R}^2$ where $P$ denotes the union of the interior and the boundary. We first analyze some general properties of detour maxima, then we develop an algorithm for the Euclidean metric, and finally, we present a faster algorithm for the $L^1$-detour of monotone rectilinear polygons.

## 2 Properties of Maxima

Ebbers-Baumann et al. [3] showed for the Euclidean norm that every polygonal chain $C$ in $\mathrm{R}^2$ has a co-visible[1] *detour maximum* $(p, q)$, i.e. $\delta_C(p, q) = \delta(C)$. The proof can easily be extended to polygons with detour $\delta(P) > 1$ and arbitrary norms.[2]

**Lemma 1** *Let $P \subset \mathrm{R}^2$ be a simple polygon with $\delta(P) > 1$ where the detour is measured with respect to an arbitrary norm $\|.\|$. Then, there always exists a detour maximum $(p, q) \in P \times P$ which is co-visible in $P^{\mathbf{C}}$* [3].

[1] In this setting, $(p, q)$ is *co-visible* iff $\overline{pq} \cap C = \{p, q\}$.
[2] Note that for Euclidean distances $\delta(P) = 1$ iff $P$ is convex.
[3] $P^{\mathbf{C}} = \mathrm{R}^2 \setminus P$ ; $(p, q)$ is *co-visible in* $P^{\mathbf{C}}$ iff $\overline{pq} \cap P = \{p, q\}$.

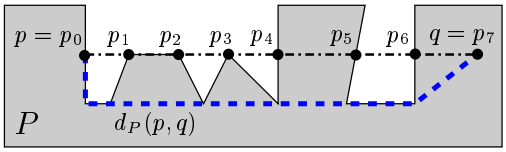


Figure 1: Boundary intersection points of $\overline{pq}$

**Proof.** Let $(p, q)$ be a detour maximum. Due to $\delta(p, q) = \delta(P) > 1$, $(p, q)$ cannot be co-visible in $P$. If $(p, q)$ is co-visible in $P^{\mathbf{C}}$, the proof is done. Otherwise, let $p_0 := p$, $p_n := q$ and let $p_1, \ldots, p_{n-1}$ be the boundary intersection points of $\overline{pq}$ apart from $p, q$ (see Fig. 1), i.e. $p_i \in \overline{pq} \cap \partial P \setminus \{p, q\}$ and $p_i$ touches $\overline{pq} \cap P^{\mathbf{C}}$.

If the points $p_i$ are ordered by their distance to $p$, we get $\|\overline{pq}\| = \sum_{i=0}^{n-1} \|\overline{p_i p_{i+1}}\|$. Additionally applying the triangle inequality of $d_P(.,.)$ yields:

$$(1) \qquad \begin{aligned} \delta_P(p, q) &= \frac{d_P(p, q)}{|\overline{pq}|} \overset{\Delta\text{-inequ.}}{\leq} \frac{\sum_{i=0}^{n-1} d_P(p_i, p_{i+1})}{\sum_{i=0}^{n-1} \|\overline{p_i p_{i+1}}\|} \\ &\leq \max_{0 \leq i \leq n-1} \frac{d_P(p_i, p_{i+1})}{\|\overline{p_i p_{i+1}}\|} = \max_{0 \leq i \leq n-1} \delta_P(p_i, p_{i+1}) \end{aligned}$$

The maximum on the right hand side is attained by a pair of points being co-visible in $P^{\mathbf{C}}$. □

For the Euclidean norm one can even show that every detour maximum of any non-convex polygon must be co-visible in $P^{\mathbf{C}}$. For the $L^1$-norm we will give an stricter statement in section 4.

# 3 Euclidean Detour of Simple Polygons

In this section, we introduce an algorithm which computes the exact Euclidean detour of a given polygon $P$ with $n$ vertices. Lemma 1 already allows us to restrict the search for detour maxima to the boundary of $P$. The following lemma further reduces the number of candidates.

**Lemma 2** *Any simple polygon $P \subset \mathrm{R}^2$ has a detour maximum $(p, q)$ which is a* vertex-boundary cut*, i.e. at least one of the points $p$, $q$ is a vertex and the other one lies on the boundary $\partial P$.*

Lemma 2 suggests the following strategy: For every vertex $p$ of $P$ and every edge $e$ of the boundary compute the local maximum $\max_{q \in e} \delta_P(p, q)$ and return the maximum of these values. However, this does not lead directly to a quadratic upper time bound because the local maximum cannot be found in constant time.

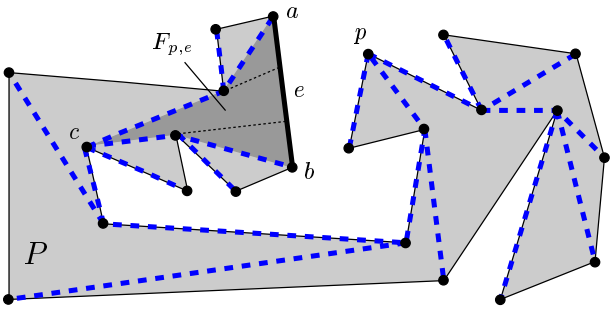


Figure 2: Shortest path tree $\mathrm{SPT}(p)$, funnel $F_{p,e}$ and its regions

To find a local maximum we consider the *funnel* $F_{p,e}$ of $p$ and $e$ (see Fig. 2) first examined by Lee and Preparata [7]. Let $a$ and $b$ be the vertices incident to $e$, then $F_{p,e}$ is the polygon bounded

by $e$ and the shortest paths $\pi(a,c)$ and $\pi(b,c)$, where $c$ is the first common vertex of $\pi(a,p)$ and $\pi(b,p)$. This vertex $c$ is called the *cusp* of the funnel, and both paths $\pi(a,c)$ and $\pi(a,b)$ are outward convex (see [5]).

For every point $q \in e$ the shortest path $\pi(q,p)$ can be divided into $\pi(q,c)$ and $\pi(c,p)$, the first one completely contained within $F_{p,e}$. We associate with $q$ the first vertex of $F_{p,e}$ hit by $\pi(q,p)$. Thus, if $k$ is the number of vertices of $F_{p,e}$, the edge $e$ will be divided into $k$ regions $R_1, \ldots, R_k$ including the degenerate cases $R_1 := \{a\}$ and $R_k := \{b\}$ (see Fig. 2). For each such region a local maximum can be computed in $O(1)$ if $F_{p,e}$ and $|\pi(p,c)|$ are known.

Hence, a local maximum of any point $p$ and any edge $e$ can be computed in $O(k)$ where $k$ is the number of vertices (or edges) of $F_{p,e}$. The funnel $F_{p,e}$ can easily be computed from the shortest path tree $\mathrm{SPT}(p)$ in $O(k)$ time by looking for the first common vertex of $\pi(a,p)$ and $\pi(b,p)$. Because every edge of the shortest path tree $\mathrm{SPT}(p)$ (see Fig. 2) can be at most on the boundary of two funnels and $\mathrm{SPT}(p)$ has $n-1$ edges, we get the value $\max_{q \in \partial P} \delta_P(p,q)$ in time $O(n)$ if $\mathrm{SPT}(p)$ is known.

Guibas et al. [5] have shown how to construct $\mathrm{SPT}(p)$ in linear time in any triangulated simple polygon. Since we can use Chazelle's [2] well-known algorithm to triangulate $P$ in linear time, our idea leads to an algorithm computing $\max_{q \in \partial P} \delta_P(p,q)$ in $O(n)$. Thus, applying Lemma 2 yields a way to get the detour of $P$ in $O(n^2)$.

**Theorem 3** *Let $P \subset \mathrm{R}^2$ be a simple polygon with $n$ vertices. Its maximum Euclidean detour value $\delta(P)$ and a pair of points $(p,q)$ attaining the maximum can be computed in time $O(n^2)$.*

However, this result might not be best possible. One can transfer the approximation algorithm of Ebbers-Baumann et al. [3] to the setting of simple polygons achieving a $(1+\varepsilon)$-approximation in $O(n \log n)$. This hints that there could be a sub-quadratic solution. The complete proofs of the previous results can be found in [4].

# 4 $L^1$-Detour of Monotone Rectilinear Polygons

Within the simpler setting of monotone rectilinear[4] polygons, we can compute the $L^1$-detour in linear time. The main reason is a stricter statement about detour maxima proven similarly to Lemma 1:

**Lemma 4** *Let $P \subset \mathrm{R}^2$ be a simple rectilinear polygon and let $(p,q) \in P \times P$ be a $L^1$-detour maximum. If $R(p,q)$ denotes the* bounding rectangle[5] *of $p$ and $q$, its intersection with $P$ must be empty apart from $p$ and $q$, i.e. $R(p,q) \cap P = \{p,q\}$.*

It follows immediately that any $L^1$-detour maximum $(p,q)$ must either be a pair of vertices or an axis-parallel pair of boundary points. In both cases, $(p,q)$ must be co-visible in $P^{\mathbf{C}}$. If $P$ is $x$-monotone[6] ($y$-monotone), further arguments yield that every maximum must be a horizontal (vertical) vertex-boundary cut.

W.l.o.g. let $P$ be $x$-monotone. We describe an algorithm examining all upper maximum candidates, i.e. horizontal vertex-boundary cuts of the upper boundary which are co-visible in $P^{\mathbf{C}}$. The lower boundary can be treated in the same way.

The algorithm starts at the left-most vertex of the upper boundary and proceeds to the right (see Fig. 3). While moving on the boundary chain, a stack holds every previously visited *left vertical segment* $s$ (i.e. $s$ is vertical and $P$ lies to the left of $s$) for which there has not been found any opposite right segment, yet. If the current boundary point is moving upward a vertical (right) edge, the algorithm pops the corresponding left segments and examines a horizontal pair each time it pops a vertex of a left segment or finds a vertex on the current right segment.

[4] A polygon is *rectilinear* iff every edge is either horizontal or vertical.
[5] $R(p,q) := \{r \in \mathrm{R}^2 \,|\, \min(p_x,q_x) \le r_x \le \max(p_x,q_x) \wedge \min(p_y,q_y) \le r_y \le \max(p_y,q_y)\}$.
[6] $P$ is *$x$-monotone* iff its intersection with any vertical line is connected.

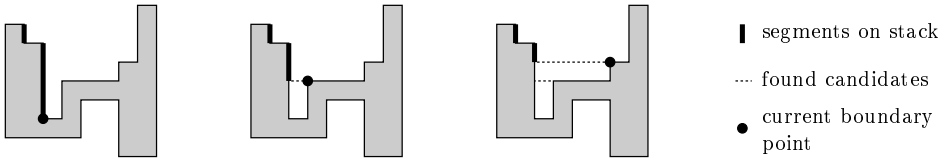


Figure 3: Some states of the algorithm for monotone rectilinear polygons

When the algorithm has found a maximum candidate $(p, q)$, it has to calculate its detour. Let $\pi(p, q)$ be a rectilinear shortest path connecting $p$ and $q$ within $P$. The $y$-length $l_y(\pi(p, q))$ is the summed up length of all vertical segments of $\pi(p, q)$. The $x$-length $l_x(\pi(p, q))$ is defined analogously. Obviously, $d_P^{L^1}(p, q) = l_x(\pi(p, q)) + l_y(\pi(p, q))$ where $l_x(\pi(p, q)) = |p_x - q_x|$ due to $P$ being $x$-monotone. Thus, for computing $\delta_P^{L^1}(p, q)$ we just need the coordinates of $p$ and $q$ and the value $l_y(\pi(p, q))$. The additional path information for calculating $l_y(\pi(p, q))$ can also be stored on the stack without increasing the linear time bound of the algorithm. Further details are omitted in this abstract.

**Theorem 5** *Let $P \subset \mathrm{R}^2$ be an $x$-monotone (or $y$-monotone) rectilinear polygon with $n$ vertices. An $L^1$-detour maximum $(p, q)$ and its value $\delta_P^{L^1}(p, q) = \delta^{L^1}(P)$ can be computed in time $O(n)$.*

# 5 Open Questions

One main questions remains open: Is there a sub-quadratic algorithm computing exactly the Euclidean detour of any simple polygon or is there a quadratic lower bound? The same problem is not solved for the presumably easier setting of simple rectilinear polygons and the $L^1$-norm.

# References

[1] P. K. Agarwal, R. Klein, C. Knauer, and M. Sharir. Computing the detour of polygonal curves. Technical report, Freie Universität Berlin, Fachbereich Mathematik und Informatik, 2002.

[2] B. Chazelle. Triangulating a simple polygon in linear time. *Discrete Comput. Geom.*, 6(5):485–524, 1991.

[3] A. Ebbers-Baumann, R. Klein, E. Langetepe, and A. Lingas. A fast algorithm for approximating the detour of a polygonal chain. *ESA 2001 - European Symposium on Algorithms*, 2001.

[4] A. Gruene. Umwege in Polygonen. Diplomarbeit, Universität Bonn, 2002.

[5] L. J. Guibas, J. Hershberger, D. Leven, M. Sharir, and R. E. Tarjan. Linear-time algorithms for visibility and shortest path problems inside triangulated simple polygons. *Algorithmica*, 2:209–233, 1987.

[6] S. Langerman, P. Morin, and M. A. Soss. Computing the maximum detour and spanning ratio of planar paths, trees and cycles. *STACS 2002*, pages 250–261, 2002.

[7] D. T. Lee and F. P. Preparata. Euclidean shortest paths in the presence of rectilinear barriers. *Networks*, 14:393–410, 1984.

[8] G. Narasimhan and M. Smid. Approximating the stretch factor of Euclidean graphs. *SIAM J. Comput.*, 30:978–989, 2000.