Shortest Paths in Polygons

Wolfgang Mulzer

1 Properties of Shortest Paths

We are given a simple polygon P, and two points s and t inside P. Let $V(P)$ be the vertex set of P, and $E(P)$ the edge set. We assume that no three points from $V(P) \cup \{s, t\}$ are on a line. We would like to find a shortest path from s to t that lies inside P, i.e., a shortest polygonal chain π with endpoints s and t such that $\pi \subseteq P$. Let $V(\pi)$ denote the vertices and $E(\pi)$ the edges of π. Every vertex in $V(\pi) \setminus \{s, t\}$ is called an *inner vertex* of π.

Lemma 1.1. *There is a shortest path π from s to t so that all inner vertices of π are vertices of P.*

Proof. Let π be a shortest path from s to t with a minimum number of inner vertices not in $V(P)$. If no such vertices exist, we are done. Thus, suppose there is $v \in V(\pi) \setminus (V(P) \cup \{s, t\})$, and let $e_1, e_2 \in E(\pi)$ be the two edges of π incident on v. If e_1 and e_2 are collinear, then v can be deleted from π, contradicting the fact that the number of inner vertices is minimum.

Suppose that v is in the interior of an edge $f \in E(P)$. There is an $\varepsilon > 0$ such that the ε-ball $B = B(v, \varepsilon)$ around v intersects only v, e_1, e_2, and the interior of f. Let w_1 and w_2 be the intersections of ∂B with e_1 and e_2. Since B intersects the polygon boundary only in f, the line segment w_1w_2 lies in P. Also, since e_1 and e_2 are not collinear, the segment w_1w_2 is strictly shorter than the chain w_1vw_2. Thus, replacing w_1vw_2 by the segment w_1w_2 yields a strictly shorter path from s to t, a contradiction to π being a shortest path; see Fig. 1.

The case that v lies in the interior of P is analogous. □

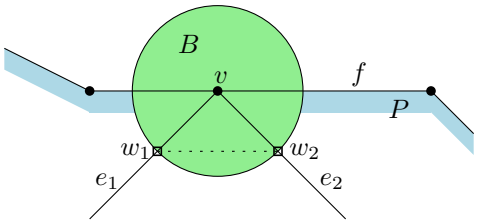

Fig. 1: If v lies in the interior of f, we can shortcut π along w_1w_2.

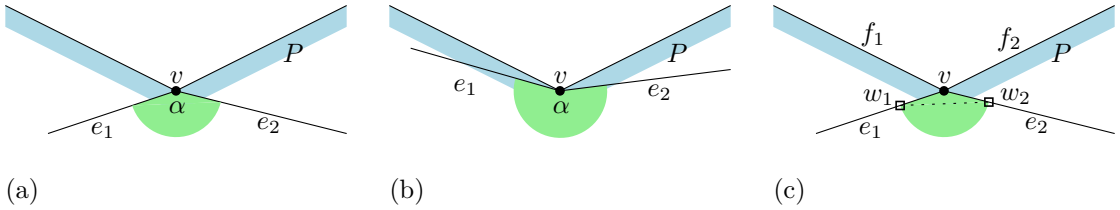

Fig. 2: The inner angle α of π at v: (a) v is not reflex; (b) v is reflex; (c) if v is not reflex, we can shortcut π at w_1w_2.

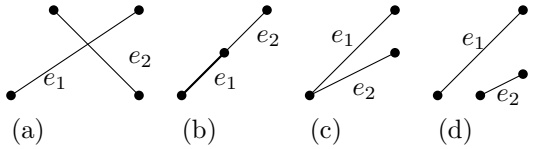

Fig. 3: (a) and (b): segments e_1 and e_2 intersect properly; (c) and (d): segments e_1 and e_2 do not intersect properly.

Now let π be a shortest path from s to t with all inner vertices in $V(P)$. Let $e_1, e_2 \in E(\pi)$ be the edges of π incident on $v \in V(P)$. The *inner angle* of π at v is the angle between e_1 and e_2 at v that lies wholly in P. We say that v is a *reflex vertex* of π if the inner angle is at least 180°, see Fig. 2(a,b).

Lemma 1.2. *Let π be a shortest path from s to t with all inner vertices in $V(P)$. All inner vertices of π are* reflex.

Proof. Suppose π has an inner vertex v that is not reflex. Let $e_1, e_2 \in E(\pi)$ be the edges of π and $f_1, f_2 \in E(P)$ the edges of P incident on v. There exists $\varepsilon > 0$ such that the ε-ball $B = B(v, \varepsilon)$ around v intersects only v, e_1, e_2, f_1, and f. Let w_1 and w_2 be the intersections of ∂B with e_1 and e_2. Since v is not reflex, the segment w_1w_2 lies in P. As in the proof of Lemma 1.1, by shortcutting at w_1w_2 we can obtain a path from s to t that is strictly shorter than π, a contradiction, see Fig. 2(c). □

Next, we show that shortest paths with the same origin do not intersect properly; see Fig. 3.

Lemma 1.3. *Let $s, t_1, t_2 \in P$. Let π_1 be a shortest path in P from s to t_1, and let π_2 be a shortest path in P from s to t_2. Suppose that all inner vertices of π_1 and π_2 lie in $V(P)$, and that no edge of π_1 and π_2 contains a vertex of $V(P)$. Then π_1 and π_2 do not intersect properly, i.e., for any two edges e_1 of π_1 and e_2 of π_2, we have either (i) $e_1 = e_2$; or (ii) the edges e_1 and e_2 are disjoint or share at most one endpoint.*

Proof. Suppose that the edges $e_1 \in E(\pi_1)$ and $e_2 \in E(\pi_2)$ intersect properly. Let x be the intersection point. Then x cannot be a vertex of P, or else e_1 or e_2 would contain a vertex

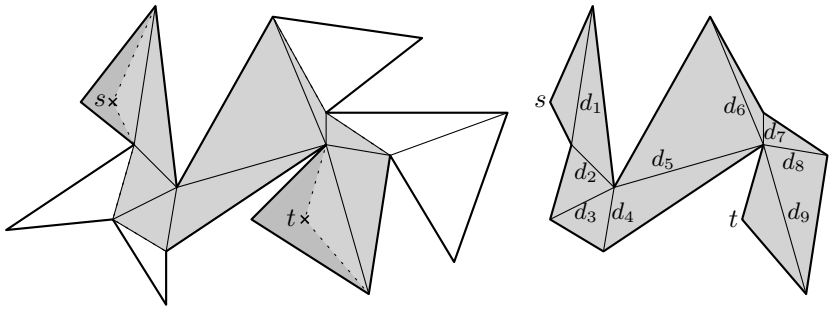

Fig. 4: (a) A triangulation of P and the triangles connecting s and t. The sleeve is shown in light gray; (b) the sleeve of s and t, with the corresponding sequence of triangles.

of P. Thus, x must lie in the interior of P. The paths $s \xrightarrow{\pi_1} x$ and $s \xrightarrow{\pi_2} x$ must have the same length; otherwise we could shorten one of π_1 or π_2. Thus, we can obtain a path from s to t_1 of the same length by taking $s \xrightarrow{\pi_2} x \xrightarrow{\pi_1} t_1$. However this path has x as an inner vertex at which the adjacent edges are not collinear. Thus, as in the proof of Lemma 1.1, we can obtain a strictly shorter path from s to t_1, a contradiction. □

Using our results so far, we can prove that shortest paths in polygons are unique.

Lemma 1.4. *The shortest path from s to t in P is unique, up to subdivision of edges.*

Proof. Exercise. □

2 Computing Shortest Paths

We now describe how to compute the shortest path from s to t in P. Let T be a triangulation of P, and let D_s be the triangle that contains s and D_t the triangle that contains t. We may assume that $D_s \neq D_t$, since otherwise the shortest path is the line segment st.

The dual graph T^* of T is a tree (exercise). Therefore, T^* contains a unique path $D_s = D_1, D_2, \ldots, D_k = D_t$ from D_s to D_t. For $i = 1, \ldots, k-1$, let d_i be diagonal that is common to D_i and D_{i+1}. Furthermore, let $D'_1 = \mathrm{CH}(s, d_1)$ and $D'_k = \mathrm{CH}(t, d_{k-1})$. The *sleeve* S of s and t in P is the polygon $D'_1 \cup D'_k \cup \bigcup_{i=2}^{k-1} D_i$; see Fig. 4. The shortest path from s to t lies completely in S, and it crosses each diagonal d_i exactly once. Hence, all vertices of π are vertices of S.

The boundary of S can be partitioned into two polygonal chains from s to t: the *left chain* (clockwise) and the *right chain* (counterclockwise). For each diagonal d_i, we denote the vertex of d_i on the left chain by l_i, and the vertex on the right chain by r_i. Now, the algorithm for computing the shortest path from s to t proceeds iteratively: we successively

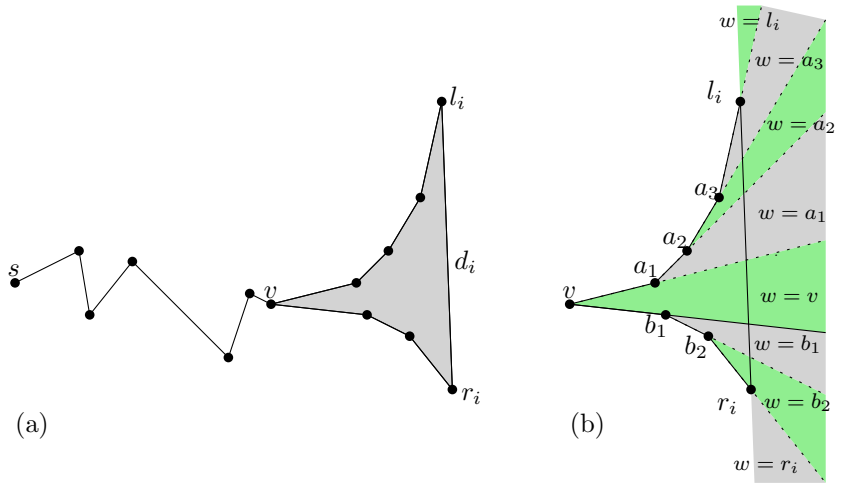

Fig. 5: (a) A funnel: v is the apex; all vertices between v and l_i lie on the left chain, all vertices between v and r_i lie on the right chain; (b) The position of r_{i+1} uniquely determines the next-to-last vertex w on ρ_{i+1}.

find the shortest paths $\lambda_1, \lambda_2, \ldots, \lambda_{k-1}$ from s to $l_1, l_2, \ldots, l_{k-1}$ and $\rho_1, \rho_2, \ldots, \rho_{k-1}$ from s to $r_1, r_2, \ldots, r_{k-1}$. The shortest paths λ_1 and ρ_1 are trivial to compute, and we will see that there is a simple method to go from λ_i and ρ_i to λ_{i+1} and ρ_{i+1}. By Lemmas 1.3 and 1.4, the paths λ_i and ρ_i share the same vertices until some vertex v, after which they diverge into two noncrossing branches.

Lemma 2.1. *Let $i \in \{1, \ldots, k-1\}$, and consider λ_i and ρ_i. Let v be the last common vertex of λ_i and ρ_i. Then, (i) all inner vertices of $v \xrightarrow{\lambda_i} l_i$ are on the left chain, and all inner vertices of $v \xrightarrow{\rho_i} r_i$ are on the right chain; (ii) $v \xrightarrow{\lambda_i} l_i$ and $v \xrightarrow{\rho_i} r_i$ are concave inside the sleeve; and (iii) $s \xrightarrow{\lambda_i} v = s \xrightarrow{\rho_i} v$.*

Proof. Exercise. □

We say that λ_i and ρ_i form a *funnel*. The path $s \xrightarrow{\lambda_i} v$ is called the *tail* of the funnel: the vertex v is the *apex* of the funnel; see Fig. 5(a). The shortest path algorithm iteratively computes the i-th funnel F_i, for $i = 1, \ldots, k-1$. How does the funnel change as we proceed from i to $i+1$? The diagonals d_i and d_{i+1} have one endpoint in common, so suppose that $l_{i+1} = l_i$ and $\lambda_{i+1} = \lambda_i$ (the other case is symmetric). We only need to find ρ_{i+1}, given F_i. Since we know the shortest paths from s to all previous vertices on the sleeve, it is enough to identify the next-to-last vertex w of ρ_{i+1}. By Lemma 1.3, the paths λ_{i+1}, ρ_i, and ρ_{i+1} do not intersect, so w must lie on F_i, on or after the apex. Finally, the inner angle of ρ_{i+1} at w must be reflex. This determines w uniquely, see Fig. 5(b).

Lemma 2.2. *The shortest path ρ_{i+1} is obtained by taking the last vertex w on ρ_i or λ_i that makes a reflex inner angle with r_i and by extending the shortest path from s over w to d^r_{i+1}. The vertex w is either the apex of F_i, or it comes after the apex.* □

3 Algorithm

We can now state the shortest path algorithm. First, we find the sleeve S for P, s, and t, and we determine the diagonals $d_1, \ldots, d_{k-1}$. The shortest paths λ_i and ρ_i are represented as two doubly linked lists that store bidirectional pointers to the vertices on the paths. Furthermore, we maintain a pointer v to the apex of the current funnel. We initialize λ_1 as $s \to l_1$ and ρ_1 as $s \to r_1$. The initial apex v is s.

In the main loop, we iterate for $i = 1, \ldots, k-2$. We take z to be the endpoint of d_{i+1} that is not an endpoint d_i. If $z = r_{i+1}$, we walk backwards along ρ_i until we encounter either v or the first vertex w that is reflex with respect to z. If w and z make a reflex vertex, we set ρ_{i+1} to $s \xrightarrow{\rho_i} w \to z$ by changing the pointers at w. Otherwise, we walk forward along λ_i up to the first vertex w that can see z; see Fig. 5(b). We then set ρ_{i+1} to $s \xrightarrow{\lambda_i} w \to z$, and we move the apex v to w; see Fig. 6. If z is l_{i+1}, we proceed symmetrically.

Once λ_{k-1} and ρ_{k-1} are known, we perform one more update step to find the shortest path from s to t.

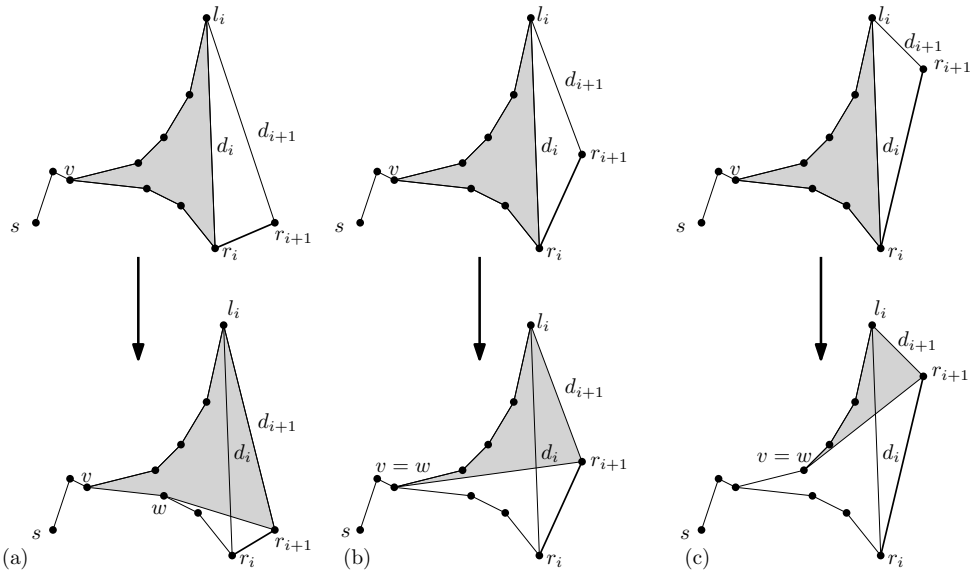

Fig. 6: Advancing the funnel: (a) w lies on the right chain; (b) w is the apex; or (c) w lies on the left chain and the apex moves forward.

4 Analysis

Correctness of the algorithm follows from Lemma 2.2. It remains to bound the running time. Using the polygon triangulation algorithm from class, we need $O(n \log n)$ time to find the

sleeve. If a triangulation of P is known, or if we use a more advanced triangulation algorithm, the time reduces to $O(n)$. Furthermore, it takes $O(1)$ steps to initialize the funnel.

Finally, we claim that the total time to update the funnel is $O(n)$: in each iteration, the running time is proportional to the number of vertices we traverse on λ_i and ρ_i. Every vertex we traverse, except the last one, is either deleted or moved behind the apex. Once this happens, this vertex is never traversed again. Thus, the amortized time per iteration is constant, and the total time is linear.

Theorem 4.1. *Given a simple polygon P with n vertices, and two points s and t in P, we can find the shortest path from s to t in P in total time $O(n)$.*