# Approximate Shortest Paths in Simple Polyhedra

Fajie Li[1] and Reinhard Klette[2]

[1]College of Computer Science and Technology, Huaqiao University, Xiamen, Fujian, China

[2]Computer Science Department, The University of Auckland, New Zealand

email: `li.fajie@hqu.edu.cn`

## Introduction

Minimal paths in volume images have raised interest in computer vision and image analysis (for example, [4, 5]). In medical image analysis, minimal paths were extracted in 3D images and applied to virtual endoscopy [5]. The existed approximation algorithms for 3D ESP calculations are not efficient, see, for example, [2, 6]. Recently, [1] proposes algorithms for calculating approximate ESP*s* amid a set of convex obstacles. For latest results related to surface ESPs, see [3]. In this paper, we apply a rubberband algorithm to present an approximate

$$\kappa(\varepsilon) \cdot \mathcal{O}(M|V|) + \mathcal{O}(M|E| + |S| + |V| \log |V|)$$

algorithm for ESP calculations when $\Pi$ is a (type-2, see Definition 2 below) simply connected polyhedron which is not necessarily convex.

The given algorithm solves approximately three NP-complete or NP-hard 3D ESP problems in time $\kappa(\varepsilon) \cdot \mathcal{O}(k)$, where $k$ is the number of layers in a stack, which is introduced as the *problem environment* below. Our algorithm has straightforward applications for ESP problems when analyzing polyhedral objects (e.g., in 3D imaging), or for 'flying' over a polyhedral terrain.

## Basics

We denote by $\Pi$ a *simple polyhedron* in the 3D Euclidean space, which is equipped with an $xyz$ Cartesian coordinate system. Let $E$ be the set of edges of $\Pi$; $V = \{v_1, v_2, \ldots, v_n\}$ the set of vertices of $\Pi$. For $p \in \Pi$, let $\pi_p$ be the plane which is incident with $p$ and parallel to the $xy$-plane. The intersection $\pi_p \cap \Pi$ is a finite set of simple polygons; a singleton is considered to be a degenerate polygon.

**Definition 1** *A simple polygon $P$, being a connected component of $\pi_p \cap \Pi$, is called a* critical polygon *of $\Pi$ (with respect to $p$).*

**Definition 2** *We say that a simple polyhedron $\Pi$ is a* type-1 *polyhedron iff any vertex $p$ defines exactly one convex critical polygon. We say that a simple polyhedron $\Pi$ is a* type-2 *polyhedron iff any vertex $p$ defines exactly one simple critical polygon.*

## ESP Computation

**Procedure 1** (compute a sequence of vertices of the critical polygon; see Fig. 1)
*Input:* Set $\mathcal{F}$ and a vertex $v \in V$ such that $\pi_v$ intersects $\Pi$ in more than just one point.
*Output:* An ordered sequence of all vertices in $V_v$, which is the vertex set of the critical polygon $P_v$.

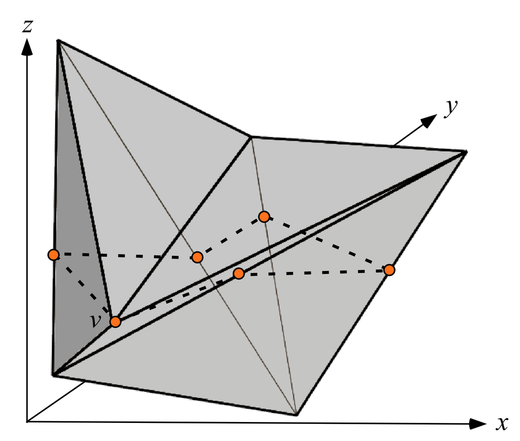


**Figure 1: The labeled vertex $v$ identifies a sequence of six vertices of the critical polygon $P_v$, defined by the intersection of plane $\pi_v$ with the shown (Schönhardt) polyhedron.**

The main ideas of the rubberband algorithm (Algorithm 1) are as follows: For a start, we randomly take a point in the closure of each critical polygon to identify an initial path from $p$ to $q$. Then we enter a loop; in each iteration, we optimize locally the position of point $p_1$ by moving it within its critical polygon, then of $p_2$, ..., and finally of $p_k$. At the end of each iteration, we check the difference between the length of the current path to that of the previous one; if it is less than a given accuracy threshold $\varepsilon > 0$ then we stop. Otherwise, we go to the next iteration.

**Algorithm 1** (a rubberband algorithm for type-1 polyhedra)
*Input:* Two points $p$ and $q$, a set $\{P^\bullet_{v_1}, P^\bullet_{v_2}, \ldots, P^\bullet_{v_k}\}$, where $P_{v_i}$ is a critical polygon of a given polyhedron $\Pi$, $k$ vertices $v_i \in \partial P_{v_i}$ such that $p_z < v_{1_z} < \cdots < v_{k_z} < q_z$, for $i = 1, 2, \ldots, k$, and there is no any other critical polygon of $\Pi$ between $p$ and $q$; given is also an accuracy constant $\varepsilon > 0$.
*Output:* The set of all vertices of an approximate shortest path which starts at $p$, then visits approximate optimal positions $p_1, p_2, \ldots, p_k$ in that order, and finally ends at $q$.

**Algorithm 2** (a rubberband algorithm for type-2 polyhedra)

1. For $i \in \{1, 2, \ldots, k\}$, apply (e.g.) the Melkman algorithm for computing $C(P_{v_i})$, the convex hull of $P_{v_i}$.
2. Let $C(P^\bullet_{v_1}), C(P^\bullet_{v_2}), \ldots, C(P^\bullet_{v_k})$, $p$, and $q$ be the input of Algorithm 1 for computing an approximate shortest route $\langle p, p_1, \ldots, p_k, q\rangle$.
3. For $i = 1, 2, \ldots, k-1$, find a point $q_i \in C(P^\bullet_{v_i})$ such that $d_e(p_{i-1}, q_i) + d_e(q_i, p_{i+1}) = \min\{d_e(p_{i-1}, p) + d_e(p, p_{i+1}) : p \in C(P^\bullet_{v_i})\}$. Update the path for each $i$ by $p_i = q_i$.
4. Let $P^\bullet_{v_1}, P^\bullet_{v_2}, \ldots, P^\bullet_{v_k}$, $p$ and $q$ be the input of Algorithm 1, and points $p_i$ as obtained in Step 3 are the initial vertices $p_i$ in Step 1 of Algorithm 1. Continue with running Algorithm 1.
5. Return $\langle p, p_1, \ldots, p_{k-1}, p_k, q\rangle$ as provided in Step 4.

**Algorithm 3** (main algorithm)
*Input:* Two points $p$ and $q$ in $\Pi$; sets $\mathcal{F}$ and $V$ of faces and vertices of $\Pi$, respectively.
*Output:* The set of all vertices of an approximate shortest path, starting at $p$ and ending at $q$, and contained in $\Pi$.

1. Initialize $V' \leftarrow \{v : p_z < v_z < q_z \wedge v \in V\}$.
2. Sort $V'$ according to the $z$-coordinate.
3. We obtain $V' = \{v_1, v_2, \ldots, v_{k'}\}$ with $v_{1z} \le v_{2z} \le \ldots \le v_{k'z}$.
4. Partition $V'$ into pairwise disjoint subsets $V_1$, $V_2$, ..., and $V_k$ such that $V_i = \{v_{i1}, v_{i2}, \ldots, v_{in_i}\}$, with $v_{ij_z} = v_{ij+1_z}$, for $j = 1, 2, \ldots, n_i - 1$, and $v_{i1_z} < v_{i+11_z}$, for $i = 1, 2, \ldots, k-1$.
5. Set $u_i \leftarrow v_{i1}$, where $i = 1, 2, \ldots, k$.
6. Set $V'' \leftarrow \{u_1, u_2, \ldots, u_k\}$ (then we have that $u_{1z} < u_{2z} < \ldots < u_{kz}$).
7. **for** each $u_i \in V''$ **do**
8. Apply Procedure 1 for computing $V_{u_i}$ (i.e., a sequence of vertices of the critical polygon $P_{u_i}$).
9. **end for**
10. Set $\mathcal{F}_{step} \leftarrow \{P^\bullet_{u_1}, P^\bullet_{u_2}, \ldots, P^\bullet_{u_k}\}$.
11. Set $P \leftarrow \{p\} \cup V'' \cup \{q\}$.
12. Apply Algorithm 2 on inputs $\mathcal{F}_{step}$ and $P$, for computing the shortest path $\rho(p, q)$ inside of $\Pi$.
13. Convert $\rho(p, q)$ into the standard form of a shortest path by deleting all vertices which are not on any edge of $\Pi$ (i.e., delete $p_i$ if $p_i$ is not on an edge of $P_{u_i}$).

## Time Complexity

We have implemented a simplified version of Algorithm 1 where all $P^\bullet_{v_i}$s were degenerated to be line segments. Thousands of experimental results *indicated* that $\kappa(\varepsilon)$ does not depend on the number $k$ of segments but the value of $\varepsilon$. We selected $\varepsilon = 10^{-15}$ and $k$ was in between 4 and 20,000, the observed maximal value of $\kappa(\varepsilon)$ was 380,000. It shows that the smallest upper bound of $\kappa(\varepsilon) \ge \kappa(10^{-15}) \ge$ 380,000. In other words, the number of iterations in the while-loop can be huge even for some small value of $k$. On the other hand, all these experimental results indicated that $|L_m - L_{m+1}| \le 1{\cdot}2$, when $m > 200$ and $L$ was between 10,000 and 2,000,000. It showed that $\kappa(1{\cdot}2) \le 200$ and the relative error $|L_m - L_{m+1}|/L \le 1{\cdot}2 \times 10^{-4}$. In other words, these experiments showed that the algorithm already reached an approximate ESP with a very minor relative error after 200 iterations of the while loop; the remaining iterations were 'just' spent on improving a very small fraction of the length of the path.

## An Example

*Example.* Let $\Pi$ be a simply connected polyhedron such that each critical polygon is the complement of an axis-aligned rectangle. The Euclidean shortest path between $p$ and $q$ inside of $\Pi$ can be approximately computed in $\kappa(\varepsilon) \cdot \mathcal{O}(|V_{pq}|)$ time. Therefore, the 3D ESP problem can be approximately solved efficiently in such a special case. Finding the exact solution is NP-complete because of the following

**Theorem 1** ([7], Theorem 4) *It is NP-complete to decide wether there exists an obstacle-avoiding path of Euclidean length at most $L$ among a set of stacked axis-aligned rectangles (see Fig. 2). The problem is (already) NP-complete for the special case that the axis-aligned rectangles are all q-rectangles of types 1 or 3.*

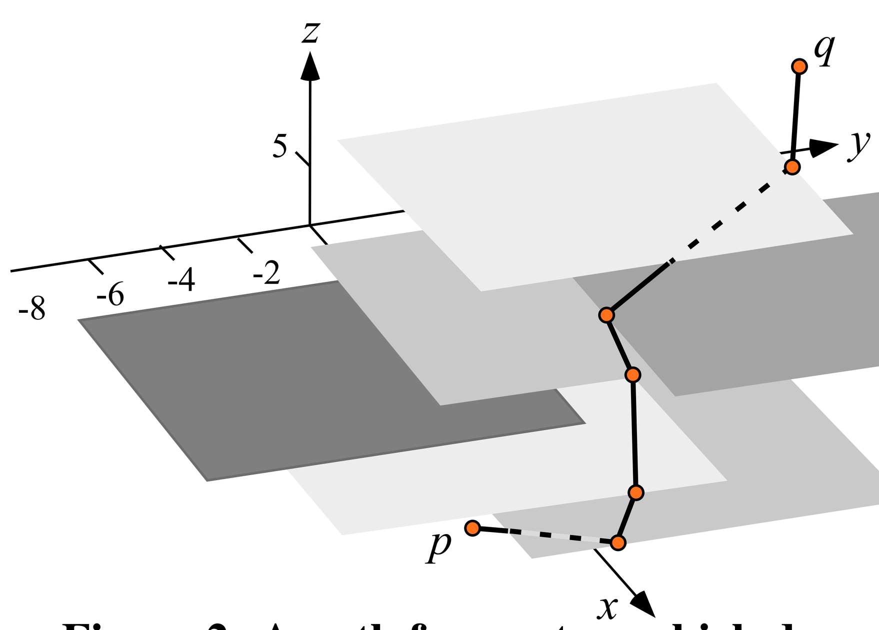


**Figure 2: A path from $p$ to $q$ which does not intersect any of the shown rectangles at an inner point.**

## Conclusions

We described an algorithm for solving the 3D ESP problem when the domain $\Pi$ is a type-2 simply connected polyhedron. Our algorithm has straightforward applications on ESP problems in 3D imaging (where proposed solutions depend on geodesics), or when 'flying' over a polyhedral terrain. As there does not exist an algorithm for finding exact solutions to the general 3D ESP problem, our method defines a new opportunity to find approximate (and efficient!) solutions to the discussed classical, fundamental, hard and general problems.

## References

[1] P. K. Agarwal, R. Sharathkumar, and H. Yu. Approximate Euclidean shortest paths amid convex obstacles. In Proc. *ACM-SIAM Sympos. Discrete Algorithms*, pages 283–292, 2009.

[2] L. Aleksandrov, A. Maheshwari, and J.-R. Sack. Approximation algorithms for geometric shortest path problems. In Proc. *ACM Sympos. Theory Comput.*, pages 286–295, 2000.

[3] M. Balasubramanian, J. R. Polimeni, and E. L. Schwartz. Exact geodesics and shortest paths on polyhedral surfaces. *IEEE Trans. Pattern Analysis Machine Intelligence*, **31**:1006–1016, 2009.

[4] F. Benmansour, L. D. Cohen. Fast object segmentation by growing minimal paths from a single point on 2D or 3D images. *J. Math. Imaging Vision*, **33**: 209–221, 2009.

[5] T. Deschamps and L. D. Cohen. Fast extraction of minimal paths in 3D images and applications to virtual endoscopy. *Med. Image Anal.*, **5**:281–299, 2001.

[6] Y. A. Liu. and S. D. Stoller. Optimizing Ackermann's function by incrementalization. In Proc. *ACM SIGPLAN Sympos. Partial Evaluation Semantics-Based Program Manipulation*, pages 85–91, 2003.

[7] J. S. B. Mitchell and M. Sharir. New results on shortest paths in three dimensions. In Proc. *ACM Sympos. Computational Geometry*, pages 124–133, 2004.