Approximate ESPs on Surfaces of Polytopes Using a Rubberband Algorithm

Fajie Li[1], Reinhard Klette[2], and Xue Fu[3,4]

[1] Institute for Mathematics and Computing Science, University of Groningen
P.O. Box 800, 9700 AV Groningen, The Netherlands
[2] Computer Science Department, The University of Auckland
Private Bag 92019, Auckland 1142, New Zealand
[3] Faculty of Economics, University of Groningen
P.O. Box 800, 9700 AV Groningen, The Netherlands
[4] School of Public Finance, Jiangxi University of Finance and Economy
Nanchang, 330013, China

Abstract. Let p and q be two points on the surface of a polytope Π. This paper provides a rubberband algorithm for computing a Euclidean shortest path between p and q (a so-called *surface ESP*) that is contained on the surface of Π. The algorithm has $\kappa_1(\varepsilon) \cdot \kappa_2(\varepsilon) \cdot \mathcal{O}(n^2)$ time complexity, where n is the number of vertices of Π, $\kappa_i(\varepsilon) = (L_{0i} - L_i)/\varepsilon$, for the true length L_i of some shortest path with initial (polygonal path) length L_{0i} (used when approximating this shortest path), for $i = 1, 2$. Rubberband algorithms follow a straightforward design strategy, and the proposed algorithm is easy to implement and thus of importance for applications, for example, when analyzing 3D objects in 3D image analysis, such as in biomedical or industrial image analysis, using 3D image scanners.

Keywords: *rubberband algorithm, Euclidean shortest path, surface ESP.*

1 Introduction

Let Π be a connected polyhedral domain such that its frontier is a union of a finite number of triangles. An *obstacle* is a connected, bounded polyhedral component in the complement $\mathbb{R}^3 \backslash \Pi$ of Π. Let $p, q \in \Pi$ such that $p \neq q$. The *general Euclidean shortest-path problem* (ESP) asks to find a shortest polygonal path $\rho(p, q)$ which is either completely contained in Π, or just not intersecting any (topologic) interior of a finite number of given obstacles.

This problem is actually a special case of the problem of planning optimal collision-free paths for a robot system; for its specification and a first result, see [13]. This paper presented in 1984 a doubly exponential time algorithm for solving the general obstacle avoidance problem. [12] improved this by providing a singly exponential time algorithm. The result was further improved by a PSPACE algorithm in [2]. Since the general ESP problem is known to be NP-hard [1], special cases of the problem have been studied afterwards. [14] gave a

polynomial time algorithm for ESP calculations for cases where all obstacles are convex and the number of obstacles is small. [4] solved the ESP problem with an $\mathcal{O}(n^{6k-1})$ algorithm assuming that all obstacles are vertical "buildings" with k different values for height.

[13] is the first publication considering the special case that the shortest polygonal path $\rho(p, q)$ is constrained to stay on the surface of Π. [13] presented an $\mathcal{O}(n^3 \log n)$ algorithm where Π was assumed to be convex. [11] improved this result by providing an $\mathcal{O}(n^2 \log n)$ algorithm for the surface of any bounded polyhedral Π. The time complexity was even reduced to $\mathcal{O}(n^2)$ [3]. So far, the best known result for the surface ESP problem is due to [10]; it improved in 1999 the time complexity to $\mathcal{O}(n \log^2 n)$, assuming that there are $\mathcal{O}(n)$ vertices and edges on Π.

This paper provides a rubberband algorithm (RBA) for computing approximate surface ESP. The algorithm has

$$\kappa_1(\varepsilon) \cdot \kappa_2(\varepsilon) \cdot \mathcal{O}(n^2)$$

time complexity, where n is the number of vertices of Π, and

$$\kappa_i(\varepsilon) = \frac{L_{0i} - L_i}{\varepsilon}$$

for the true length L_i of some kind of shortest path with length L_{0i} of the used initial polygonal path, for $i = 1, 2$.

Although this rubberband algorithm is not the most efficient, it follows a straightforward design strategy, and the proposed algorithm is easy to implement. (See [8] for results on implementing rubberband algorithms for various shortest path problems.)

We generalize a rubberband algorithm from solving the 2D ESP of a simple polygon (see [9] for this 2D algorithm) to a solution for the surface ESP of polytopes. Considering the difficulty of the general ESP problem, our approach is very important for applications, e.g. when analyzing 3D objects in 3D image analysis (such as in biomedical or industrial image analysis, using 3D image scanners). For shortest paths on digital surfaces (in the context of 3D picture analysis), also known as *geodesics*, see the monograph [6]. One of the earlier publications, related to the calculation of surface geodesics, is [5].

The rest of this paper is organized as follows: Section 2 provides the definitions of some useful notions. Section 3 presents four procedures being subroutines of the Main Algorithm. Section 4 proves the correctness of the Main Algorithm. Section 5 analyses the time complexities for involved procedures and Main Algorithm. Section 6 illustrates the main ideas behind the steps of the Main Algorithm by a simple example. Section 7 summarizes the paper.

2 Definitions

Let Π be a polytope (see Figure 1 for an example). Let $T = \{\triangle_1, \triangle_2, \ldots, \triangle_m\}$ be a set of triangles such that $\partial\Pi = \cup_{i=1}^{m} \triangle_i$ and $\triangle_i \cap \triangle_j = \emptyset$ or $= e_{ij}$ or $= v_{ij}$,

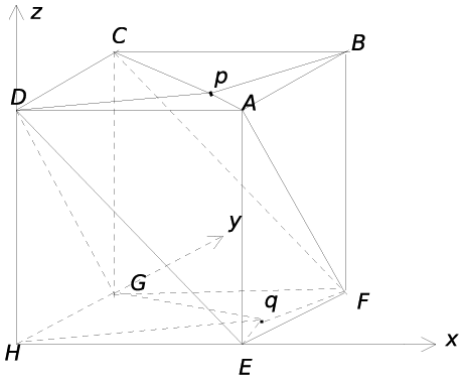

Fig. 1. An unit cube within the xyz-coordinate system, where $p = (0.76, 0.12, 1)$, $q = (0.9, 0.24, 0)$.

where e_{ij} (v_{ij}) is an edge (vertex) of both $\triangle_i$ and $\triangle_j$, $i \neq j$, respectively, with $i, j = 1, 2, \ldots, m$.

We construct a corresponding simple graph $G_{\Pi} = [V_{\Pi}, E_{\Pi}]$ where $V_{\Pi} = \{v_1, v_2, \ldots, v_m\}$. Each v_i is a triangle. Edges $e \in E_{\Pi}$ are defined as follows: If $\triangle_i \cap \triangle_j = e_{ij} \neq \emptyset$, then we have an edge $e = v_i v_j$ (where e_{ij} is an edge of both triangles $\triangle_i$ and $\triangle_j$); and if $\triangle_i \cap \triangle_j = \emptyset$ or a vertex, then there is not an edge between v_i and v_j, $i < j$ and $i, j = 1, 2, \ldots, m$.

In such a case we say that G_{Π} is a *corresponding graph* with respect to the triangulated polytope Π. See Figure 2 for an example. Analogously, we can define a corresponding graph for a connected surface segment (a *subsurface*) of a polytope. Abbreviated, we may also speak about "the graph for a polytope" or "the graph for a subsegment of a surface".

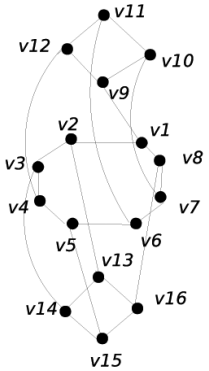

Fig. 2. A (3-regular) corresponding graph of the polytope in Figure 1.

A triangulated polytope Π can also be thought as being a graph such that each vertex of Π is a vertex of this graph, and each edge of a triangle is an edge of this graph. We denote this graph by G'_Π.

Let $p \neq q$, p, $q \in V(G'_\Pi)$; if ρ is a cycle of G'_Π such that $G'_\Pi \backslash \rho$ has two components, denoted by G_1 and G_2 with $p \in V(G_1)$ and $q \in V(G_2)$, then ρ is called a *cut cycle* of G'_Π or Π. For example, in Figure 1, $ABCDA$ or $AFGDA$ are cut cycles of Π.

An *approximate* cycle is a graph such that it consists of a cycle plus a few more vertices, each of which is of degree one only, and (thus) adjacent to a vertex on the cycle. (The graph later shown in Figure 4 is an approximate cycle.)

A *band* is a subsurface of a polytope Π such that the corresponding graph of it is a cycle or an approximate cycle.

A band can also be thought as being a subgraph of G'_Π. Let E' be the subset of all the edges of a triangulated band such that each edge belongs to a unique triangle. Then E' consists of two cycles. Each of them is called a *frontier* of the band. For example, in Figure 1, $ABCDA$ and $EFGHE$ are two frontiers of a band whose triangles are perpendicular to the xoy-plane.

If two triangulated bands share a common frontier, then they are called *continuous* (in the sense of "continuation").

3 Algorithms

Without loss of generality, we can assume that $p \neq q$, p and $q \in V(\Pi)$.

3.1 Separation

The following procedure finds a cut cycle to separate p and q such that either p or q is not a vertex of the cut cycle. (This procedure will be used in Step 1 of the Main Algorithm below.)

Procedure 1

Input: $G'_\Pi = [V(\Pi), E(\Pi)]$, and two vertices $p \neq q \in V(\Pi)$.

Output: The set of all vertices of a cycle ρ in G such that, if we cut the surface of Π along ρ into two separated parts, then p and q are on different parts respectively.

1. Let $N_p = \{v : vp \in E(\Pi)\}$ (i.e., the set of all neighbors of p).
2. Select u, $v \in N_p$ such that $\angle upv \neq 180°$. In other words, $uv \in E(\Pi)$.
3. Let $V = \{p, v\}$.
4. Let $N_v = \{w' : w'v \in E(\Pi)\}$ (i.e., the set of all neighbors of v).
5. Take a vertex $w \in N_v \backslash V$.
6. If $w = u$, then stop. Otherwise, let $V = V \cup \{w\}$, $v = w$ and go to Step 4.
7. If $q \notin V$, then output V. Otherwise, output $V \backslash \{q\}$.

For example, in Figure 1, ρ can be either $ABCDA$ or $AFGDA$, but it can not be $AEHDA$.

3.2 Step Set Calculation

The following procedure computes step bands (i.e., the step set for the second level RBA). It will be used in Step 2 of the Main Algorithm below.

Procedure 2

Input: $G'_{\Pi} = [V(\Pi), E(\Pi)]$, and ρ: the cut cycle obtained with Procedure 1. Without loss of generality, we can assume that $p \in V(\rho)$ and $q \notin V(\rho)$.

Output: The set of the step bands $S = \{B_1, B_2, \ldots, B_m\}$ such that $p \in V(B_1)$ and $q \in V(B_m)$.

1. Let $S = \emptyset$, $\Pi_1 = \Pi$ and $\rho_1 = \rho$.

2. While $q \notin V(\rho_1)$, do the following:

2.1. Let $\Pi_2 = \Pi_1 - \rho_1$ such that $q \in V(\Pi_2)$. (Note: the used "minus" in graph theory can also be written as $\Pi_1 \backslash \rho_1$; in other words, we delete each vertex in ρ_1 and each edge of Π_1 which is incident with a vertex of ρ_1.)

2.2. Let ρ_2 be the frontier of Π_2.

2.3. Let Π_1, ρ_1 and ρ_2 as the input, compute a band $B = G_{\Pi_1}(V(\rho_1) \cup V(\rho_2))$ (the induced subgraph of G_{Π_1}).

2.4. Update ρ_1 and Π_1 by letting $\rho_1 = \rho_2$ and $\Pi_1 = \Pi_2$.

2.5. Let $S = S \cup \{B\}$.

3. Output S.

For example, in Figure 1, if a single vertex can be thought of as being a band, then we can have $S = \{B_1, B_2, B_3\}$, where $B_1 = p$, B_2 is the band such that $V(B_2) = \{A, B, C, D, E, F, G, H\}$, and $B_3 = q$.

3.3 Step Segments in a Single Band

The following procedure computes step segments in a single band (i.e., a subset of the step set for the initialization of the RBA). (It will be used in Step 1.1 of Procedure 4 below.)

Procedure 3

Input: The triangulated band B and two vertices u, $v \in V(B)$ such that u and v are on two different frontiers of B, denoted by ρ_1 and ρ_2 (i.e., $u \in V(\rho_1)$ and $v \in V(\rho_2)$.

Output: Two step sets of segments (edges) S_1 and S_2 such that either S_1 or S_2 contains the vertices of a surface ESP of B from u to v.

Let $\triangle_u$, $\triangle_v$ be the triangles such that $u \in \partial\triangle_u$ and $v \in \partial\triangle_v$, respectively. Let w_u and $w_v \in V(G_B)$ such that w_u and w_v correspond to $\triangle_u$ and $\triangle_v$, respectively.

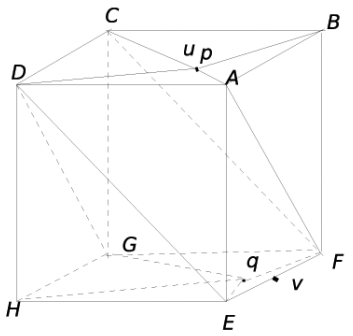

Fig. 3. A unit cube such that $u = p$, and v is the center of EF.

By the definition of a band (see Section 2), there is a cycle, denoted by ρ_B, such that either w_u (respectively, w_v) $\in V(\rho_B)$ or the unique neighbor of w_u (respectively, w_v) is in $V(\rho_B)$.

For example, in Figure 3, the frontier of B consists of two cycles $uABCDu$ and $EFGHE$. We have that $\triangle_u = \triangle pDA$, $\triangle_v = \triangle AEF$. $S_1 = \{AD, AE\}$ and $S_2 = \{DA, DE, DH, DG, CG, CF, BF, AF\}$.

Case 1: Both w_u and w_v are in $V(\rho_B)$. In this case, ρ_B can be decomposed into two paths from w_u and w_v, denoted by P_1 and P_2. Let $\{\triangle_1, \triangle_2, \ldots, \triangle_{m_1}\}$ be the sequence of triangles corresponding to the sequence of the vertices of P_1.

Let $\{e_1, e_2, \ldots, e_{m_1-1}\}$ be a sequence of edges such that $e_i = \triangle_i \cap \triangle_{i+1}$, where $i = 1, 2, \ldots, m_1 - 1$.

Let $\{e'_1, e'_2, \ldots, e'_{m_1-1}\}$ be a sequence of edges such that e'_i is obtained by removing a sufficiently small segment (Assume that the length of the removed segment equals δ'.) from both endpoints of e_i, where $i = 1, 2, \ldots, m_1 - 1$.

The set $\{e'_1, e'_2, \ldots, e'_{m_1-1}\}$ is the approximate step set we are looking for.

Case 2: Both w_u and w_v are not in $V(\rho_B)$. Again, by the definition of a band (see Section 2), let w'_u (w'_v) be the unique neighbor of w_u (w_v) such that w'_u and $w'_v \notin V(\rho_B)$.

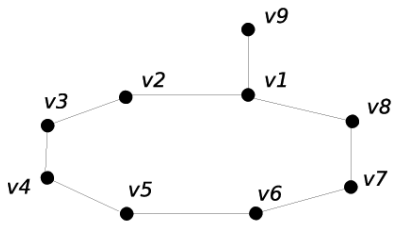

Fig. 4. The corresponding graph with respect to B; the two frontiers of B are $pABCDp$ and $EFGHE$ in Figure 3. $v9$ corresponds to $\triangle pDA$, and $v2$ corresponds to $\triangle AEF$.

In this case, ρ_B can be decomposed into two paths from w'_u and w'_v, denoted by P'_1 and P'_2. Appending w_u and w_v to both ends of P'_1 and P'_2, we obtain two paths, denoted by P_1 and P_2. Analogous to Case 1, we can compute the approximate step set.

Case 3: Only one of either w_u or w_v is not in $V(\rho_B)$. We can compute the approximate step set, analogously to Cases 1 and 2.

3.4 Initializations

The following procedure is the initialization procedure of the RBA. It will be used in Steps 7.2 and 8.2 of the Main Algorithm below.

Procedure 4

Input: Two continuous triangulated bands B_1 and B_2, and three vertices u_1, u_2 and u_3, all three in $V(B_1 \cup B_2)$, such that u_1 and u_2 are on two different frontiers of B_1, denoted by ρ_1 and ρ_2; u_3 is on the frontier denoted by ρ_3 ($\neq \rho_2$), of B_2.

Output: The set of vertices of an approximate ESP on the surface of $B_1 \cup B_2$, from u_1 to u_3.

Let $e_{u_2} \in E(\rho_2)$ such that $u_2 \in e_{u_2}$; l a sufficiently large integer; and $E = E(\rho_2)$.

1. While $E \neq \emptyset$, do the following:

1.1. Let G_{B_i} and u_i, u_{i+1} be the input; apply Procedure 3 to compute step segments in band B_i, denoted by S_{B_i}, where $i = 1, 2$.

1.2. Let $S_{12} = S_{B_1} \cup S_{B_2}$ be the input. Apply Algorithm 1 of [9] to compute an approximate ESP on the surface of $B_1 \cup B_2$. This is denoted by $\rho_{e_{u_2}}$, and it connects u_1 with u_3.[1]

1.3. Let the length of $\rho_{e_{u_2}}$ be equals $l(\rho_{e_{u_2}})$.

1.4.1. If $l(\rho_{e_{u_2}}) < l$ then let $V = V(\rho_{e_{u_2}})$.

1.4.2. If $l(\rho_{e_{u_2}}) = l$ then let $V = \min_{lexi}\{V, V(\rho_{e_{u_2}})\}$ (minimum with respect to lexicographic order).

1.5. Let $E = E \backslash \{e_{u_2}\}$.

1.6. Take an edge $e \in E$ and let u_2 be one endpoint of e; let $e_{u_2} = e$; go to Step 1.1.

2. Output V.

3.5 The Main Algorithm

The main algorithm defines now the iteration steps of the RBA.

[1] Note that Algorithm 1 of [9] still works when the line segments in the step set are in 3D space.

Input: $G'_{\Pi} = [V(\Pi), E(\Pi)]$, and two vertices $p \neq q$, p, $q \in V(\Pi)$; accuracy constant ε.

Output: The set of vertices of an approximate ESP on the surface of Π.

1. Let G'_{Π}, p and q be the input; apply Procedure 1 to compute a cut cycle which separates p and q, denoted ρ_{pq}.

2. Let G'_{Π} and ρ_{pq} be the input; apply Procedure 2 to compute step bands $S = \{B_1, B_2, \ldots, B_m\}$ such that $p \in V(B_1)$ and $q \in V(B_m)$.

3. Let p_i be a point on the frontier of B_i, where $i = 1, 2, \ldots, m$, $p = p_1$ and $q = p_m$. We obtain an initial path $\rho =< p_1, \ldots, p_2, \ldots, p_m >$ [note: it is very likely that there exist further points between p_i and p_{i+1}!].

The following steps are modified from Algorithm 1 of [9] (note: only Steps 7.2 and 8.2 are modified!).

4. Let $\varepsilon = 10^{10}$ (the chosen accuracy).

5. Compute the length L_1 of the initial path ρ.

6. Let $q_1 = p$ and $i = 1$.

7. While $i < k$ - 1 do:

7.1. Let $q_3 = p_i + 1$.

7.2. Apply Procedure 4 to compute a point q_2 on the frontier of B_i such that q_2 is a vertex of an approximate ESP on the Surface of $B_{i-1} \cup B_i$ from q_{i-1} to q_{i+1}.

7.3. Update ρ by replacing p_i by q_2 [possibly also by some additional points between p_{i-1} and p_i, and between p_i and p_{i+1}!].

7.4. Let $q_1 = p_i$ and $i = i + 1$.

8.1. Let $q_3 = q$.

8.2. Apply Procedure 4 to compute a point q_2 on the frontier of B_m such that q_2 is a vertex of an approximate ESP on the surface of $B_{m-1} \cup B_m$, from q_{m-1} to q_{m+1}.

8.3. Update ρ by replacing p_k by $q - 2$ [note: possibly also by additional points between p_{m-1} and p_m, and between p_m and p_{m+1}!].

9. Compute the length L_2 of the updated path ρ.

10. Let $\delta = L_1$ - L_2.

11. If $\delta > \varepsilon$, then let $L_1 = L_2$ and go to Step 7. Otherwise, stop.

We provide a proof of correctness, an analysis of run-time complexity, and an example for this algorithm. It is basically another illustration for the general comments (e.g., in [7, 8]) that the basic idea of rubberband algorithms my be applied efficiently for a diversity of shortest path problems.

4 Proof of Correctness

Theorem 1. *The approximate path computed by the Main Algorithm is an approximate ESP on the surface of Π.*

Proof. Let $\{B_1, B_2, \ldots, B_m\}$ be the step bands computed by Step 2 of the Main Algorithm. Let $\rho_i = B_i \cap B_{i+1}$, where $i = 1, 2, \ldots, m-1$. For each point $p_i \in \rho_i$, where $i = 1, 2, \ldots, m-1$, the length of the surface path

$$\rho =< p_1, \ldots, p_2, \ldots, p_m >$$

is a continuous function defined on $\rho_1 \cdot \rho_2 \cdot \ldots \cdot \rho_{m-1}$, denoted by $\Pi_{i=1}^{m-1} \rho_i$.

Since each ρ_i is a closed set, where $i = 1, 2, \ldots, m-1$, $\Pi_{i=1}^{m-1} \rho_i$ is a closed set as well.

Since ρ is continuous, for each $\varepsilon > 0$ and for each $P = (p_1, p_2, \ldots, p_{m-1}) \in \Pi_{i=1}^{m-1} \rho_i$, there exists a $\delta_\varepsilon > 0$, such that for each $P' \in U(P, \delta_\varepsilon)$, the difference between the lengths (i.e., of the outputs) of the Main Algorithm by using either P or P' as an initial path, is not more than ε.

We can now construct an open cover for $\Pi_{i=1}^{m-1} \rho_i$ as follows:

$$O_\varepsilon = \{U(P, \delta_\varepsilon) : P \in \Pi_{i=1}^{m-1} \rho_i\}$$

By the finite cover principle of mathematical analysis, there exists a finite subcover of O_ε which can cover $\Pi_{i=1}^{m-1} \rho_i$. This implies that the number of approximate ESPs obtained by the Main Algorithm is finite. In analogy to the proof of Lemma 24 of [7], the number of approximate ESPs obtained by the Main Algorithm is only one. This proves the theorem.

5 Time Complexity

This section analyses, step by step, the time complexity for each of the procedures and the Main Algorithm as presented in the previous section.

Lemma 1. *Procedure 1 can be computed in time $\mathcal{O}(|V(\Pi)|^2)$.*

Proof. In our data structure we identify adjacent vertices for each vertex; so Steps 1 and 4 can be computed in time $\mathcal{O}(|V(\Pi)|)$. Step 2 can be computed in time $\mathcal{O}(|N_p|)$. Step 3 can be computed in time $\mathcal{O}(1)$. Step 5 can be computed in time $\mathcal{O}(|N_v|)$. Step 6 can be computed in time $\mathcal{O}(1)$. The loop, from Step 4 to Step 6, is computed in time $\mathcal{O}(|V(\Pi)|^2)$. Step 7 can be computed in time $\mathcal{O}(|V|)$. Therefore, Procedure 1 can be computed in time $\mathcal{O}(|V(\Pi)|^2)$.

Lemma 2. *Procedure 2 can be computed in time $\mathcal{O}(|\Pi_1|^2)$.*

Proof. Step 1 can be computed in time $\mathcal{O}(1)$. The test in Step 2 can be computed in time $\mathcal{O}(|V(\rho_1)|)$. Step 2.1 can be computed in time $\mathcal{O}(|V(\Pi_1)|)$. Step 2.2 can be computed in time $\mathcal{O}(|V(\Pi_2)|)$. Step 2.3 can be computed in time $\mathcal{O}(|V(\Pi_1)|)$. Steps 2.4 and 2.5 can be computed in time $\mathcal{O}(1)$. The loop, from Step 2 to Step 2.5, is computed in time $\mathcal{O}(|\Pi_1| \cdot |V(\rho_1)|) \leq \mathcal{O}(|\Pi_1|^2)$. Step 3 can be computed in time $\mathcal{O}(|S|)$. Therefore, Procedure 2 can be computed in time $\mathcal{O}(|\Pi_1|^2)$.

The following Lemma is obvious. Step 1.4.1 can be computed in time $\mathcal{O}(1)$.

Lemma 3. *Procedure 3 can be computed in time* $\mathcal{O}(|V(G_B)|)$.

Lemma 4. *Procedure 4 has time complexity* $\kappa_1 \cdot \mathcal{O}(|V(\rho_2)| \cdot |V(B_1 \cup B_2)|)$.

Proof. By Lemma 3, Step 1.1 can be computed in time $\mathcal{O}(|V(B_i)|)$, where $i = 1, 2$. By Theorem 1.4 of [9], Step 1.2 has time complexity

$$\kappa_1 \cdot \mathcal{O}(|V(\rho_2)| \cdot |V(B_1 \cup B_2)|)$$

where $\kappa_1 = (L_0 - L)/\varepsilon$, ε is the accuracy, and L_0 and L are the lengths of the initial and true path, respectively.

Step 1.3 can be computed in time $\mathcal{O}(|V(\rho_{e_{u_2}})|)$. Step 1.4.1 can be computed in time $\mathcal{O}(1)$. Step 1.4.2 can be computed in time $\mathcal{O}(|V(B_1 \cup B_2)|)$. Step 1.5 can be computed in time $\mathcal{O}(|V(\rho_2)|)$. Step 1.6 can be computed in time $\mathcal{O}(1)$. The loop, from Step 1.1 to 1.6, can be computed in time

$$\kappa_1 \cdot \mathcal{O}(|V(\rho_2)| \cdot |V(B_1 \cup B_2)|)$$

Step 2 can be computed in time $\mathcal{O}(|V|)$. Therefore, Procedure 2 can be computed in time $\kappa_1 \cdot \mathcal{O}(|V(\rho_2)| \cdot |V(B_1 \cup B_2)|)$.

Theorem 2. *The Main Algorithm has time complexity* $\kappa_1 \cdot \kappa_2 \cdot \mathcal{O}(|V(G_\Pi)|^2)$.

Proof. By Lemma 1, Step 1 can be computed in time $\mathcal{O}(|V(G_\Pi)|^2)$. According to Lemma 2, Step 2 can be computed in time $\mathcal{O}(|V(G_\Pi)|^2)$. Step 3 can be computed in time $\mathcal{O}(|V(\rho)|)$. Steps 4, 6, 10 and 11 can be computed in time $\mathcal{O}(1)$. Steps 5 and 9 can be computed in time $\mathcal{O}(|V(\rho)|)$. Steps 7.1, 7.4 , 8.1 can be computed in time $\mathcal{O}(1)$. By Lemma 4, Steps 7.2 and 8.2 can be computed in time $\kappa_1 \cdot \mathcal{O}(|V(\rho_{j2})| \cdot |V(B_{i-1} \cup B_i)|)$, where $j = i, m$. Steps 7.3 and 8.3 can be computed in time $\mathcal{O}(|V(\rho)|)$. The loop, from Step 7 to 11, can be computed in time $\kappa_1 \cdot \kappa_2 \cdot \mathcal{O}(|V(G_\Pi)|^2)$.

Therefore, the Main Algorithm can be computed in time $\kappa_1 \cdot \kappa_2 \cdot \mathcal{O}(|V(G_\Pi)|^2)$.

6 An Example

The following example illustrates the steps of the Main Algorithm. Let Π be the unit cube in Figure 5.

Step 1 computes a cut cycle (which may be not uniquely defined) $\rho_{pq} = ABCDA$.

Step 2 computes step bands $S = \{B_1, B_2, B_3\}$, where $B_1 = p$, B_2's frontiers are two cycles $ABCDA$ and $EFGHE$, and $B_3 = q$.

Step 3 decides that we use $pIJq$ as an initial surface path from p to q (see Figure 5).

In Step 7.2, the algorithm applies Procedure 4 (the initialization procedure of the RBA) and searches each edge of the polygon $ABCDA$; it finds a point

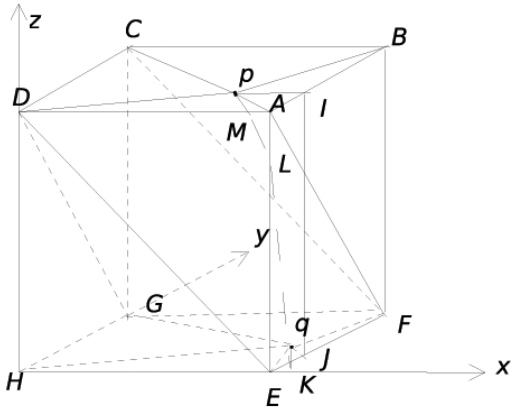

Fig. 5. A unit cube within an xyz-coordinate system, where $p = (0.76, 0.001, 1)$, $q = (0.999, 0.001, 0)$. $pIJq$ is an initial surface path from p to q while $pMLKq$ is an approximate surface ESP from p to q, where $I \in AB$, $J, K \in EF$, $L \in AE$ and $M \in AD$.

$M \in AD$ to update the initial point I, and it also inserts a new point $L' \in AE$ into the segment between M and J.

Analogously, in Step 8.2, the algorithm searches each edge of the polygon $EFGHE$ and finds a point $K \in EF$ for updating the initial point J; it also updates point $L' \in AE$ by point $L \in AE$ which is between M and K.

The algorithm iterates (note: the iteration steps are defined in the Main Algorithm) until the required accuracy is reached.

7 Conclusions

The paper presented a rubberband algorithm for computing an approximate surface ESP of a polytope. Although it is not the most efficient, it follows a straightforward design strategy, and is thus easy to implement.

This algorithm generalized an rubberband algorithm designed for solving a 2D ESP of a simple polygon (see [9]) to one which solves the surface ESP of polytopes. This approach is a contribution towards the exploration of efficient approximate algorithms for solving the general ESP problem. This will allow more detailed studies of computer-represented surfaces as typical (e.g.) in biomedical or industrial 3D image analysis.

Acknowledgement: The authors thank the PSIVT reviewers whose comments have been very helpful for revising an earlier version of this paper.

References

1. J. Canny and J. H. Reif. New lower bound techniques for robot motion planning problems. In Proc. *IEEE Conf. Foundations Computer Science*, pages 49–60, 1987.
2. J. Canny. Some algebraic and geometric configurations in PSPACE. In Proc. *ACM Symp. Theory Computation*, pages 460–467, 1988.
3. J. Chen and Y. Han. Shortest paths on a polyhedron. In Proc. *ACM Symp. Computational Geometry*, pages 360–369, 1990.
4. L. P. Gewali, S. Ntafos, and I. G. Tollis. Path planning in the presence of vertical obstacles. Technical Report, Computer Science, University of Texas at Dallas, 1989.
5. N. Kiryati and G. Szekely. Estimating shortest paths and minimal distances on digitized three dimensional surfaces. *Pattern Recognition*, **26**:1623–1637, 1993.
6. R. Klette and A. Rosenfeld. *Digital Geometry*. Morgan Kaufmann, San Francisco, 2004.
7. F. Li and R. Klette. Exact and approximate algorithms for the calculation of shortest paths. Report 2141, IMA, The University of Minnesota, Minneapolis, 2006.
8. F. Li and R. Klette. Rubberband algorithms for solving various 2D or 3D shortest path problems. Plenary Talk, In Proc. *Computing: Theory and Applications*, Platinum Jubilee Conference of The Indian Statistical Institute, pages 9 - 18, IEEE, 2007.
9. F. Li and R. Klette. Euclidean shortest paths in simple polygons. Technical Report CITR-202, Computer Science Department, The University of Auckland, Auckland, 2007. See *http://www.citr.auckland.ac.nz/techreports/2007/CITR-TR-202.pdf*.
10. S. Kapoor. Efficient computation of geodesic shortest paths. In Proc. *ACM Symp. Theory Computation*, pages, 770–779, 1999.
11. J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou. The discrete geodesic problem. *SIAM J. Computation*, **16**:647–668, 1987.
12. J. H. Reif and J. A. Storer. A single-exponential upper bound for shortest paths in three dimensions. *J. ACM*, **41**:1013–1019, 1994.
13. M. Sharir and A. Schorr. On shortest paths in polyhedral spaces. *SIAM J. Computation*, **15**:193–215, 1986.
14. M. Sharir. On shortest paths amidst convex polyhedra. *SIAM J. Computation*, **16**:561–572, 1987.