Sequences of Spheres in $\mathbb{R}^3$

Fajie Li* Gisela Klette† Reinhard Klette‡

Abstract

One version of the Euclidean shortest path problem (ESP) is to find the shortest path such that it starts at p and ends at q and it avoids passing through an unordered set of pairwise disjoint obstacles. This paper describes an approximate algorithm for solving this ESP problem for two points p and q and a finite set of pairwise disjoint spheres in $\mathbb{R}^3$ not containing those two points. We apply the Agarwal et al. [1] algorithm for the computation of an approximate shortest path in the free space between pairwise disjoint regular polyhedra in a preprocessing step that defines an order of the given obstacles. The resulting path is used as input for a new rubberband algorithm (RBA) that gives an approximate answer to the open question "What is the complexity of the Euclidean shortest path problem for obstacles that are disjoint balls?".

This RBA also provides a solution to the basic version of a touring-a-sequence-of-spheres problem (TSP) that finds a shortest path starting at p, visits all spheres in a given order and ends at q.

The paper discusses at first a solution for the 2-dimensional case (i.e., disks and polygons instead of spheres and polyhedra), followed by showing that this solution extends to the 3-dimensional case.

1 Introduction

Many algorithms for solving various *Euclidean shortest path* (ESP) problems consider geometric objects with piecewise linear (2D or 3D) frontier segments. For example, obstacles of ESPs are modelled by polygons in 2D in [10], by polyhedra in 3D in [1, 2, 5, 6, 7, 10, 13], and by the surface of polyhedra in 2·5D in [4, 10]. Obstacles with smooth surfaces are of interest in computational geometry [11]. In this paper, we propose an algorithm for the shortest path problem avoiding sets of unordered pairwise-disjoint disks in 2D or spheres in 3D and we analyse its complexity. Our approach does not deliver an exact solution of the problem but it is a contribution for finding an answer to the open problem: "What is the complexity of the Euclidean shortest path problem for obstacles that are disjoint balls?"; see [11]. An algorithm for finding an exact solution to the general 3D ESP problem does not always exist according to Theorem 9 in [3].

*College of Computer Science and Technology, Huaqiao University, Xiamen, Fujian, China, li.fajie@yahoo.com

†School of Computing & Mathematical Sciences, Auckland University of Technology, Private Bag 92006, Auckland 1142, New Zealand, gklette@gmail.com

‡Computer Science Department, The University of Auckland, Private Bag 92019, Auckland 1142, New Zealand, r.klette@auckland.ac.nz

2 Preliminaries

Let D be a disk, and p_1 and p_2 be two points in ∂D, denoting the frontier of D. Let $A_1(p_1, p_2)$ and $A_2(p_1, p_2)$ be the two arcs from p_1 to p_2 in ∂D. We denote by $d_D(p_1, p_2)$ the length of the shorter arc $A_D(p_1, p_2)$ between p_1 and p_2.

In $\mathbb{R}^2$, a regular m-gon P is called a *sketching* m-gon of a disk D if D is P's smallest circumscribing circle.

Analogously, let S be a sphere, and p_1 and p_2 be two points in ∂S, denoting the frontier of S. Let $A_S(p_1, p_2)$ be an arc from p_1 to p_2 in ∂S of minimum length. We denote by $d_S(p_1, p_2)$ the length of this shortest arc $A_S(p_1, p_2)$.

In $\mathbb{R}^3$, a polyhedron H is called a *sketching* polyhedron of a sphere S iff each vertex of H is in ∂S (i.e., the surface of S).

For a path ρ, let $L(\rho)$ be its length when applying the Euclidean distance function d_e.

We recall some notations from [1]. Let $\mathcal{P} = \{P_1, P_2, \ldots, P_k\}$ be a set of pairwise-disjoint convex polyhedral obstacles in $\mathbb{R}^d$, for $d = 2$ or $d = 3$. Let n be the total number of facets of obstacles in $\mathcal{P}$. The topological closure of $\mathbb{R}^d \backslash (\cup \mathcal{P})$ is called the *free space* of $\mathcal{P}$, denoted by $\mathcal{F}(\mathcal{P})$. An *ε-shortest path* between two points p and q in $\mathcal{F}(\mathcal{P})$ is a path whose length is at most $(1+\varepsilon)$ times the length of the shortest path. The value $1+\varepsilon$ is called the *approximation factor* of this approximation algorithm. We also call the path an $(1+\varepsilon)$-*approximation* path.

We make use of the following results of [1]:

1. In $\mathbb{R}^2$, for any two points $p, q \in \mathcal{F}(\mathcal{P})$ and a parameter ε with $0 < \varepsilon \leq 1$, an ε-shortest path between p and q in $\mathcal{F}(\mathcal{P})$ can be computed in $\mathcal{O}(n + (k/\sqrt{\varepsilon}) log(k/\varepsilon))$ time.

2. In $\mathbb{R}^3$, for any two points p, $q \in \mathcal{F}(\mathcal{P})$ and a parameter $0 < \varepsilon \leq 1$, an ε-shortest path between p and q in $\mathcal{F}(\mathcal{P})$ can be computed in $\mathcal{O}(n + (k^4/\varepsilon^7) log^3(k/\varepsilon))$ time.

3 Results

We describe at first an algorithm for computing an approximate Euclidean shortest path between points p and q visiting each disk of an ordered set of pairwise-disjoint disks in $\mathbb{R}^2$.

Later we apply this algorithm and the Agarwal et al. [1] algorithm to present an algorithm for computing an approximate Euclidean shortest path between points p and q visiting some disks of an unordered set of pairwise-disjoint disks in $\mathbb{R}^2$.

Finally we (straightforwardly) generalize these from 2D to algorithms that deal with a sequence of pairwise-disjoint spheres and an unordered set of pairwise-disjoint spheres in $\mathbb{R}^3$.

3.1 Algorithm in $\mathbb{R}^2$

The new algorithm for the ordered sequence of disks follows the general design principle of a *rubberband algorithm* (RBA): a set of *steps* is identified (set of arc segments that constitute an initial path between p and q) in an initial computation. The positions of all vertices of the calculated path in the previous iteration are locally optimized one by one (by length minimization) in the current iteration. An iteration is the final one if a termination criterion is satisfied [8, 9, 12].

We start with an RBA for a sequence of pairwise disjoint disks; see Fig. 1. This is a subprocedure of the main algorithm given below. The input is a set $\mathcal{P} = \{D_1, D_2, \ldots, D_k\}$ of k pairwise disjoint disks given in an order, two points p, q in free space $\mathcal{F}(\mathcal{P})$, and an accuracy constant $\varepsilon > 0$. Points p, q and all k disks are co-planar. The output is a sequence $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k_1}, p_{k_2}, q\rangle$ which starts at $p = p_0 = p_{0_1} = p_{0_2}$, then visits disks D_i at points p_{i_1} and p_{i_2} in the given order (i.e., both points are on the frontier ∂D_i and are not identical in general), and ends at $q = p_{k+1} = p_{k+1_1} = p_{k+1_2}$.

Note that the path in Fig. 1 visits each disk (TSP problem) in the given order. We only consider a pre-calculated subset of disks for the problem of finding the shortest path between p and q in the free space $\mathbb{R}^d \backslash (\cup \mathcal{D})$.

The basic idea of the main algorithm for the 2-dimensional case is quite straightforward. We employ a regular m-gon for approximating a disk such that the disk's frontier is the circumscribing circle of the m-gon. Then we apply the Agarwal et al. algorithm in $\mathbb{R}^2$ to find an ε_0-shortest path ρ between p and q avoiding those k regular m-gon obstacles. This ε_0-shortest path ρ delivers the initial steps (i.e., the frontiers of a set of ordered disks) for the RBA shown in Fig. 1. The RBA computes a new approximate shortest path ρ' avoiding the interiors of all disks. The value $L(\rho')$ is lower bounded by the length $L(\rho) - \varepsilon_0$ that is a lower bound of the length of a true ESP between p and q. The approximation factor of this algorithm is $L(\rho')/(L(\rho) - \varepsilon_0)$.

1: For each $i \in \{1, 2, \ldots, k\}$, let c_i be the centre of D_i. Let vertices p_{i-1_2}, p_{i_1} and p_{i_2}, p_{i+1_1} be the intersection points between the line segments $c_{i-1}c_i$ and $c_i c_{i+1}$ and the disk frontiers ∂D_{i-1}, ∂D_i and ∂D_{i+1} and let $p = c_0 = p_{0_2}$ and $q = c_{k+1} = p_{k+1_1}$. Initialize the path $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k_1}, p_{k_2}, q\rangle$.
2: Let $L_0 = +\infty$. Calculate $L_1 = d_e(p, p_{1_1}) + \sum_{i=1}^{k} (d_{D_i}(p_{i_1}, p_{i_2}) + d_e(p_{i_2}, p_{i+1_1}))$.
3: **while** $L_0 - L_1 \geq \varepsilon$ **do**
4:   **for** $i = 1, 2, \ldots, k$ **do**
5:     **if** $p_{i-1_2} p_{i+1_1} \cap D_i \neq \emptyset$ **then**
6:       Compute tangential points $q_{i_1} \in A_D(p_{i_1}, p_{i_2})$, and $q_{i_2} \in A_D(p_{i_1}, p_{i_2})$ such that $p_{i-1_2} q_{i_1}$ is a tangent to $A_D(p_{i_1}, p_{i_2})$ and $p_{i+1_1} q_{i_2}$ is a tangent to $A_D(p_{i_1}, p_{i_2})$
7:       Update $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k_1}, p_{k_2}, q\rangle$ by replacing p_{i_j} by t_{i_j} in this path (where $j = 1, 2$).
8:     **else**
9:       Compute a point $q_i \in A_D(p_{i_1}, p_{i_2})$ such that $d_e(p_{i-1_2}, q_i) + d_e(q_i, p_{i+1_1}) = \min\{d_e(p_{i-1_2}, p) + d_e(p, p_{i+1_1}) : p \in \partial D_i\}$
10:       Update $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k_1}, p_{k_2}, q\rangle$ by replacing p_{i_j} by q_i in this path (where $j = 1, 2$).
11:     **end if**
12:   **end for**
13:   Let $L_0 = L_1$. Calculate L_1 as in Line 2.
14: **end while**
15: Return $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k_1}, p_{k_2}, q\rangle$.

Figure 1: An RBA for ordered pairwise disjoint disks.

The main algorithm is shown in Fig. 2. The input is a set $\mathcal{P}$ of k pairwise disjoint disks $D_1, D_2, \ldots, D_k$ in the same plane π, two points p, q in $\mathcal{F}(\mathcal{P})$, two accuracy constants $\varepsilon > 0$ and $\varepsilon_0 > 0$, and an integer $m > 0$. The output is a sequence $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k'_1}, p_{k'_2}, q\rangle$ which starts at $p = p_0 = p_{0_1} = p_{0_2}$, then visits disks D'_i at points p_{i_1} and p_{i_2} in the given order, and finally ends at $q = p_{k'+1} = p_{k'+1_1} = p_{k'+1_2}$.

3.2 Algorithm in $\mathbb{R}^3$

The basic idea of our algorithm for $\mathbb{R}^3$ is the same as for the one proposed for the 2-dimensional case.

The proposed RBA for the 3D case is analogous to that one in 2D case shown in Fig. 1. The input is a sequence of k pairwise disjoint spheres in $\mathcal{P} = \{S_1, S_2, \ldots, S_k\}$, two points p, q in $\mathcal{F}(\mathcal{P})$, and an accuracy constant $\varepsilon > 0$. The output is a sequence $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k_1}, p_{k_2}, q\rangle$ which starts at point $p = p_0 = p_{0_1} = p_{0_2}$, then visits spheres S_i at

Let $m = 3$ and $L(\rho')/(L(\rho)$ be $+\infty$.
while the value of $L(\rho')/(L(\rho) - \varepsilon_0) - 1$ is not yet small enough **do**
  For each $i \in \{1, 2, \ldots, k\}$, compute a sketching m-gon P_i of D_i.
  Let $\mathcal{P} = \{P_1, P_2, \ldots, P_k\}$ and ε_0 be inputs, apply the Agarwal et al. algorithm in $\mathbb{R}^2$ [1] (see also Section 2) to compute an ε_0-short path ρ between p and q.
  Let $\mathcal{P}' = \{P'_1, P'_2, \ldots, P'_{k'}\}$ be a sequence of regular m-gons such that ρ starts at p, then crosses regular m-gons P'_i at points p_{i_1}, p_{i_2}, $\ldots$, $p_{i_{m_i}}$ $(1 \leq m_i \leq m)$ in the given order, and finally ends at q.
  Let $\mathcal{D}' = \{D'_1, D'_2, \ldots, D'_{k'}\}$ be the sequence of disks such that D'_is frontier is P'_is circumscribing circle.
  Let $\mathcal{D}'$ and ε be inputs for the RBA shown in Fig. 1 to compute a path ρ' between p and q.
  Let $\mathcal{D}'' = \{D''_1, D''_2, \ldots, D''_{k''}\}$ be the new sequence of disks such that ρ' starts at p, then intersects disks D''_i in the given order, and finally ends at q.
  while $\mathcal{D}'' \neq \mathcal{D}'$ **do**
    Let $\mathcal{D}' = \mathcal{D}''$.
    Let $\mathcal{D}'$ and ε be the input for the proposed RBA shown in Fig. 1 to compute the path ρ'' between p and q and $\mathcal{D}''$ is the new sequence of disks intersected by ρ''.
    Let $\rho' = \rho''$.
  end while
  Let $m = m + 1$.
end while
Return ρ'.

Figure 2: Main algorithm for a set of pairwise disjoint disks.

For each $i \in \{1, 2, \ldots, k\}$, compute a sketching polyhedra P_i of S_i.
$L(\rho')/(L(\rho)$ be $+\infty$.
while the value of $L(\rho')/(L(\rho) - \varepsilon_0) - 1$ is not yet small enough **do**
  Let $\mathcal{P} = \{P_1, P_2, \ldots, P_k\}$ and ε_0 as input, apply the Agarwal et al. algorithm in $\mathbb{R}^3$ [1] (see also Section 2) to compute an ε_0-short path ρ between p and q.
  Let $\mathcal{P}' = \{P'_1, P'_2, \ldots, P'_{k'}\}$ be the sequence of polyhedra connected by the computed path ρ between p and q.
  Let $\mathcal{S}' = \{S'_1, S'_2, \ldots, S'_{k'}\}$ be the sequence of spheres such that S'_is frontier is P'_is circumscribing sphere.
  Let $\mathcal{S}'$ and ε be inputs for the RBA to compute an initial path ρ' between p and q.
  Let $\mathcal{S}'' = \{S''_1, S''_2, \ldots, S''_{k''}\}$ be the new sequence of spheres crossed by straight segments of path ρ' between p and q.
  while $\mathcal{S}'' \neq \mathcal{S}'$ **do**
    Let $\mathcal{S}' = \mathcal{S}''$.
    Let $\mathcal{S}'$ and ε be inputs for the RBA to compute the path ρ'' between p and q and the new sequence $\mathcal{S}''$.
    Let $\rho' = \rho''$.
  end while
  For each $i \in \{1, 2, \ldots, k\}$, compute a refined sketching polyhedra of S_i.
end while
Return ρ'.

Figure 3: Main algorithm for a set of pairwise disjoint spheres.

points p_{i_1} and p_{i_2} in the given order, and finally ends at $q = p_{k+1} = p_{k+1_1} = p_{k+1_2}$.

The main algorithm is shown in Fig. 3. The inputs are spheres, two points p, q in $\mathcal{F}(\mathcal{P})$, two accuracy constants $\varepsilon > 0$ and $\varepsilon_0 > 0$, and an integer $m > 0$. The output is a sequence of points $\langle p, p_{1_1}, p_{1_2}, p_{2_1}, p_{2_2}, \ldots, p_{k'_1}, p_{k'_2}, q\rangle$ that specifies a path with a length only an approximation factor apart from the Euclidean shortest path avoiding all spheres between p and q.

4 Analysis

Regarding the time complexity of the 2-dimensional RBA shown in Fig. 1, note that the main computation is in the two stacked loops. The while-loop takes $\kappa(\varepsilon)$ iterations, where $\kappa(\varepsilon) = \frac{L_0 - L}{\varepsilon}$; L_0 and L are the lengths of initial path and output path, respectively. Lines 5–12 can be computed in $\mathcal{O}(1)$. Thus, the for-loop can be computed in time $\mathcal{O}(k)$. Thus, the algorithm can be performed in time $\kappa(\varepsilon) \cdot \mathcal{O}(k)$.

For testing the actual numerical values of $\kappa(\varepsilon)$, we have implemented a "simplified version" of the algorithm shown in Fig. 1 where all disks were degenerated to be line segments. For $\varepsilon = 10^{-15}$ and (at first "just") $k = 2$, we were running the algorithm more than 10^8 times. During those experiments, we had fixed points $p = (15, 0)$ and $q = (120, 480)$ but randomly created k pairwise disjoint line segments for each experiment. The maximum and mean values of $\kappa(\varepsilon)$ observed in those experiments are 77,170 and 4·75, respectively. We also recorded the length L_{200} of the path obtained in the 200-th iteration if $\kappa(\varepsilon) > 200$ for those experiments. We assume that the final output path is a "very good" (for common application areas) approximation path because $\varepsilon = 10^{-15}$ is already "very small". Then we obtained the approximation factor L_{200}/L of our algorithm if we terminate the algorithm after 200 iterations, where L is the length of the output "true" path. We observed that

$L_{200}/L < 1{\cdot}19$ for all our inputs. Then we also changed the value of k to be a "small" integer between 3 and 20, and performed again a large number of experiments for studying the approximation factor L_{200}/L.

The time complexity of the main algorithm in $\mathbb{R}^2$ (shown in Fig. 2) can be analysed as follows: Line 3 can be computed in $\mathcal{O}(mk)$ time. Line 4 can be computed in $\mathcal{O}(n + (k/\sqrt{\varepsilon_0})log(k/\varepsilon_0))$ time. Lines 5, 6, 8, and 16 can be computed in $\mathcal{O}(k)$ time. Lines 7 and 11 can be computed in $\kappa(\varepsilon) \cdot \mathcal{O}(k)$ time. Line 9 requires $\mathcal{O}(k^2)$ time. Lines 10, 12 and 14 can be computed in $\mathcal{O}(1)$ time. Thus, we obtain the following

Theorem 1 *The main algorithm in $\mathbb{R}^2$ can be computed in $\mathcal{O}(mk+(k/\sqrt{\varepsilon_0})log(k/\varepsilon_0)+k^3+\kappa(\varepsilon)\cdot k)$ time to obtain an $L(\rho')/(L(\rho)-\varepsilon_0)$-approximation path between p and q among pairwise disjoint disks.*

We may adjust parameters m, ε_0, ε to let the approximation factor $L(\rho')/(L(\rho)-\varepsilon_0)$ be sufficiently small and, at the same time, to terminate the while loop quickly.

The time complexity of the main algorithm in $\mathbb{R}^3$ (shown in Fig. 3) can be analysed as follows: Line 4 requires $\mathcal{O}(n + (k^4/{\varepsilon_0}^7)log^3(k/\varepsilon_0))$ time. The other lines in this algorithm can be analysed exactly the same way as those of the main algorithm in $\mathbb{R}^2$ (shown in Figure 2).

Thus, we have the following

Theorem 2 *The main algorithm in $\mathbb{R}^3$ can be computed in $\mathcal{O}(mk + (k^4/{\varepsilon_0}^7)log^3(k/\varepsilon_0) + \kappa(\varepsilon) \cdot k)$ time to obtain an $L(\rho')/(L(\rho)-\varepsilon_0)$-approximation path between p and q among pairwise disjoint spheres.*

Again, we may adjust parameters m, ε_0, ε for having the approximation factor $L(\rho')/(L(\rho) - \varepsilon_0)$ be sufficiently small and, at the same time, to terminate the while loop quickly.

5 Conclusion

We proposed an $L(\rho')/(L(\rho) - \varepsilon_0)$-approximation algorithm for computing an approximate Euclidean shortest path between two points among pairwise disjoint spheres, where $L(\rho)$ and $L(\rho')$ are the lengths of output path and the initial path, respectively, ε_0 is an accuracy parameter for the used Agarwal et al. algorithm. This algorithm provides an approximate answer to an open problem in computational geometry. The algorithm for spheres was presented by discussing at first the 2-dimensional case of disks, and then showing the analogy of a solution in 3D space.

References

[1] P. K. Agarwal, R. Sharathkumar, and H. Yu. Approximate Euclidean shortest paths amid convex obstacles. In Proc. *ACM-SIAM Symp. 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 Symp. Theory Computation*, pages 286–295, 2000.

[3] C. Bajaj. The algebraic complexity of shortest paths in polyhedral spaces. In Proc. *Allerton Conf. Communication Control Computing*, pages 510–517, 1985.

[4] 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.

[5] J. Choi, J. Sellen, and C.-K. Yap. Precision-sensitive Euclidean shortest path in 3-space. In Proc. *ACM Symp. Computational Geometry*, pages 350–359, 1995.

[6] K. L. Clarkson. Approximation algorithms for shortest path motion planning. In Proc. *ACM Symp. Theory Computing*, pages 56–65, 1987.

[7] S. Har-Peled. Constructing approximate shortest path maps in three dimensions. In Proc. *ACM Symp. Computational Geometry*, pages 125–130, 1998.

[8] F. Li, and R. Klette. Watchman route in a simple polygon with a rubberband algorithm. In Proc. *Canadian Conf. Computational Geometry*, pages 1–4, Winnipeg, Canada, 2010.

[9] F. Li and R. Klette. Euclidean Shortest Paths. Springer-Verlag. ISBN 978-1-4471-2255-5. http://www.springerlink.com/content/t11235/#section=981144&page=1

[10] J. S. B. Mitchell. Geometric shortest paths and network optimization. In *Handbook of Computational Geometry* (J.-R. Sack and J. Urrutia, editors), pages 633–701, Elsevier, 2000.

[11] J. S. B. Mitchell and M. Sharir. New results on shortest paths in three dimensions. In Proc. *SCG*, pages 124–133, 2004.

[12] X. Pan, F. Li, and R. Klette. Approximate shortest path algorithms for sequences of pairwise disjoint simple polygons. In Proc. *Canadian Conf. Computational Geometry*, pages 175–178, Winnipeg, Canada, 2010.

[13] C. H. Papadimitriou. An algorithm for shortest path motion in three dimensions. *Inform. Processing Letters*, **20**:259–263, 1985.