Between a Rock and a Hard Place:

Euclidean TSP in the Presence of Polygonal Obstacles

A Thesis

Submitted to the Faculty

of

Drexel University

by

Jeff Abrahamson

in partial fulfillment of the

requirements for the degree

of

Master of Science in Computer Science

May 2005

Acknowledgements

Special thanks to Pawel Winter for suggesting the problem and to Ali Shokoufandeh for his guidance in approaching the problem and in preparing this thesis and the papers that lead to it. Thanks are due also to Dave Richardson, Trip Denton, Fatih Demirci, and John Novatnack. Finally, the anonymous referees of the papers leading up to this thesis made many helpful suggestions that improved not only the papers they had reviewed but this thesis as well.

Table of Contents

List of Figures iv
Abstract vii
1. Overview 1
1.1 Introduction 1
1.2 Background and Related Work 1
1.2.1 Complexity 1
1.2.2 Brute Force Solutions 2
1.2.3 Approximation Solutions 2
1.2.4 Variations, Exact Cases, and Related Problems 2
1.2.5 Applications 3
1.3 Contribution 4
1.4 Structure of this Thesis 4
2. Notation and Definitions 5
3. Arbitrary Non-convex Obstacles 8
3.1 Structural Properties 8
3.2 Algorithm 13
3.3 Complexity 16
4. Convex Obstacles and Other Shortcuts 17
5. Non-planar Tours 18
6. Application: Navigation and Path Planning 21
7. Conclusions 25
Bibliography 26

List of Figures

2.1 The detour $d^k_{i,j}$ from p_k to p^+_k through $q_i \rightsquigarrow q_j$. 6

2.2 A TSP tour crosses a pocket A of Convex(Q). 7

3.1 The construction from Proposition 1. 8

3.2 Lemma 2: Replacing p_1p_3 and p_2p_4 with p_1vp_2 and p_3vp_4 results in a tour of the same length. Relaxing to $p_1p_5p_2$ and $p_3p_6p_4$ results in a shorter tour. 9

3.3 Lemma 2, case 2: three vertices on P, using a detour across Q. The bold dashed line shows a shorter tour section than the original non-bold dashed tour section. 10

3.4 Lemma 2, case 3: one vertex on P. The bold dashed line shows a shorter tour section than the original non-bold dashed tour section. 11

3.5 A shortest tour might visit some P and Q vertices more than once. 11

3.6 In the special case that P is convex, the case that the optimal tour of $P \cup$ Convex(Q) crosses a pocket. 13

3.7 In the special case that P is convex, the case that the optimal tour of $P \cup$ Convex(Q) does not cross some pocket is no easier than the general case of two arbitrary polygons. 14

3.8 A pocket with $O(m)$ peaks and $O(n)$ vertices in P above.. 14

5.1 The construction of non-nested 3-D polytopes P and Q in oblique and side views. The traveling salesman tour (red) is free to visit any vertex of P and Q in any order as long as the vertices are visible to each other. By construction, the tour visits the vertices of G (blue) separately from the non-G vertices. 19

6.1 A triangular obstacle inside another triangle. 21

6.2 The shortest path graph from a triangular obstacle inside another triangle. The solid lines indicate the $c_{i,j}$, the dotted lines indicate the zero weight return edges. 22

6.3 The shortest path visiting the vertices of two triangles. 22

6.4 An irregular corridor. 23

6.5 The shortest tour along the polygonal channel formed by an irregular corridor. Note that we form the channel by wrapping the ends around so that they form two polygons. 23

6.6 The shortest tour along an irregular corridor. 23

6.7 The shortest tour of watchpoints around a building (with service garage) and the perimeter of its property. 24

pour aller nulle part

et pourtant j'en suis fier.

– Yves Duteil, "Le petit pont de bois"

Abstract

Between a Rock and a Hard Place:
Euclidean TSP in the Presence of Polygonal Obstacles

Jeff Abrahamson
Advisor: Ali Shokoufandeh, Ph.D.

I give an $O(n^2m+nm^2+m^2 \log m)$ time and $O(n^2+m^2)$ space algorithm for finding the shortest traveling salesman tour through the vertices of two simple polygonal obstacles in the Euclidean plane, where n and m are the number of vertices of the two polygons.

1. Overview

1.1 Introduction

The Euclidean TSP (ETSP) is the problem of finding a tour of minimum length through a given set of points in d-dimensional Euclidean space. In this paper, I address a variant of the ETSP in which the points are the vertices of two polygons P and Q with n and m vertices respectively. I assume the tour may not cross P or Q. I give an $O(n^2m + nm^2 + m^2 \log m)$ time and $O(n^2 + m^2)$ space algorithm for finding the shortest tour T through the vertices of P and Q while completely avoiding crossing any edge or vertex of Q or P.

In a paper with Ali Shokoufandeh and Pawel Winter [1] we previously considered a simpler version of this problem in which P and Q are both convex. We gave an $O(m^2 \log m + m^2n)$ time and $O(nm + m^2)$ space exact solution to that problem.

1.2 Background and Related Work

For an excellent introduction to the TSP in its many forms, I refer the reader to [20, 15]. What follows is an extraordinarily abbreviated tour of notable highlights in this rich field that, in the capricious opinion of the author, are most related to this treatise.

1.2.1 Complexity

The general TSP problem considers a complete weighted graph on n points and asks for the least weight connected tour (equivalently, connected 2-factor). The problem is NP-hard [12, 17]. Moreover, it is not even approximable to any polynomial bound [31].

The restricted problem in which the weight matrix is a Euclidean distance matrix is trivially 2-approximable by considering shortcuts from a doubled (overlayed) minimum spanning tree [3]. Christofides [5] improved this bound to 1.5, a bound that stood for many years and was only recently broken by Aurora and others. In the Euclidean case the quadrangle inequality allows a proof that optimal tours do not self-intersect. (In the general graph theoretic formulation, intersection other than at a vertex is not meaningful anyway.)

1.2.2 Brute Force Solutions

Brute force solution suffers from a combinatorial explosion. Branch-and-Bound search simplifies the search somewhat, amounting to exploring the TSP polytope of the linear programming problem. Unfortunately, the LP formulation in general has exponentially many constraints [14]. Chvatal's cutting plane technique further reduces the search space [2].

1.2.3 Approximation Solutions

Aurora suspects that no PTAS is likely to exist for metric TSP unless P=NP [13]. Trevisan has shown that no PTAS exists for Euclidean TSP in greater than $O(\log n)$ dimensions [30]. Aurora, however, showed a $(1+\epsilon)$-approximation with $n^{O(1/\epsilon)}$ running time, later improved to $n(\log n)^{O(1/\epsilon)}$, for geometric TSP. Rao and Smith improved on this to $O(n \log n + n/\text{poly}(\epsilon))$. Some of these algorithms have bearing on the related problems of k-TSP, Steiner Tree, and k-MST. Johnson and Lindenstrauss showed that if a PTAS exists in $O(\log n)$ dimensions, then P=NP [15, 18].

1.2.4 Variations, Exact Cases, and Related Problems

TSP as well as k-TSP, MST, and k-MST come in bottleneck versions as well, also NP-hard. The max-TSP problem [16] inverts the optimization goal.

Relaxing the TSP to allow sub-tours (that is, the problem of finding a minimum weight, not necessarily connected, 2-factor on a graph) is also known as the assignment problem [21] and is solvable in polynomial time. Motivated by the assignment problem, we note in passing that the TSP may be expressed as an integer programming problem. Integer programming, however, is NP-hard [24].

The time-dependent TSP considers time-varying weights on the graph's edges [25]. The Traveling Repairman Problem [11] weights early visits more favorably than later visits.

Cutler [6] gave an $O(n^3)$ time and $O(n^2)$ space dynamic programming algorithm for solving the *3-line* ETSP where all points lie on three distinct parallel lines in the plane. Deineko et al. [8]

considered a related variant of the ETSP with a convex polygon P and a set of points on a line segment Q inside P. They referred to this problem as the convex-hull-and-line ETSP, and gave an $O(m^2 + mn)$ time and $O(m + n)$ space algorithm, where n and m are the number of P and Q vertices, respectively. Rote [27, 26] extended this result to m-line ETSP by giving a polynomial dynamic programming algorithm for a fixed number of lines N.

Rote [9, 26] introduced the necklace condition, which he attributes to a 1968 paper of Supnick. A set of points satisfies the necklace condition if there exists a set of balls, possibly of differing radii, centered at the points, each overlapping precisely two other balls. This defines a two-factor. If the two-factor is connected, it is a TSP tour.

Deineko et al. [7] addressed the problem of recognizing instances of the ETSP for which there is a permutation of points such that the underlying distance matrix fulfills so called Demidenko, Kalmanson and Supnick properties. It is known that if a distance matrix fulfills one of these properties, the ETSP is solvable in polynomial time. Burkard et al. [4] present a summary of known conditions on distance matrices that each make the TSP solvable in polynomial time.

1.2.5 Applications

The TSP arises in forms other than the classic case of a merchant making his rounds. Job scheduling, for instance, presents a simple but non-obvious example. Consider a machine that has some state after finishing a job (the output, the mess, the required maintenance, etc.). In addition, beginning a job requires some specific state (output area possibly emptied, input provided, maintenance check list, etc.). The jobs themselves have constant costs, but the transition work—setting the machine up for a new job given the state it was in at the end of the last job—corresponds to the edges of a graph whose nodes are the jobs themselves. Scheduling a set of jobs in the most efficient order is thus an instance of the TSP.

An example closer to our particular approach is building printed circuit boards. Consider a robot arm constrained to move in the X and Y directions that must pick up a single part from a collection of bins placed around the board before placing the part on the board. The robot repeats this process until all parts are placed and the board continues to the wave soldering machine. The robot (or, rather, the robot's owner) wants to find the least cost arm path that builds the board. Since PC components typically lie on straight lines, the results of Rote [26] offer an exact polynomial time solution.

1.3 Contribution

The present work generalizes the results of [1] to the case of arbitrary simple polygons. In particular, I present a generalization of the two nested polygon ETSP, which is itself a generalization of the line and polygon ETSP problem. As one might expect, we bear a small cost for the generalization: the computational complexity increases slightly at each step.

1.4 Structure of this Thesis

After presenting definitions in Section (2), I characterize the structure of a valid tour in Section (3.1) before presenting an algorithm and proof of its correctness in Section (3.2). In Section (3.3) I prove the space and time complexity results that, together with the algorithm, are the principal contributions of this paper. In Section (4) I show a few minor additional assumptions that improve the space and time complexity of the algorithm. Finally, in Section (6) I present some applications of the present work.

2. Notation and Definitions

I begin by revealing a useful slight of hand. The problem we address is easier to picture when the polygon Q is contained entirely within the polygon P. In fact, we do not need to use that property anywhere in the proofs, neither of the algorithm's correctness nor of its space or time complexity. Nonetheless, for the sake of exposition, and especially for the sake of the reader's ease, it is convenient to assume on a first reading that the polygons are so nested. Let us agree, then, at the outset, that we will allow ourselves to entertain this fiction, even occasionally in the diagrams that accompany the text, all the while being vigilant to note that we never actually *use* that information.

That said, however, we do use the nesting property in one implicit way, which is in discussion of same and opposite orientation tours. The structural properties of an opposite orientation tour on nested polygons correspond to a same orientation tour on non-nested polygons and vice versa. The exposition in this regard implicitly assumes that Q is nested inside P. In fact, at each point where tour orientation is discussed, one must say, quite cumbersomely, "if the polygons are nested, then. . . ; otherwise, do the same thing but switch the sense of relative orientation." I beg the reader's kind indulgence for sparing both of us that (hopefully unnecessary) awkwardness.

In the following we assume a distance function that obeys the triangle inequality and is additive: $d(a, c) = d(a, b) + d(b, c)$ for all co-linear points a, b, and c with b between a and c.

Let P denote a polygon in the plane and Q a polygonal obstacle that is completely in the interior of P. Let $p_1, \ldots, p_n$ denote the vertices of P and $q_1, \ldots, q_m$ be the vertices of Q, with $m, n \geq 3$. Throughout this paper we will assume that vertices are numbered in the clockwise direction and that subscripts are interpreted modulo n for P and m for Q. We will also denote two consecutive P or Q vertices as v and v^+.

An *edge* uv that connects a pair of points u and v is a straight line segment between u and v of length $\| uv \|$. If a line segment between any points u and v avoids crossing any edge or vertex of P and Q, then the two points are said to be *visible* to each other. A *path* is a sequence $u_1, u_2, \ldots, u_k$

of vertices along with their interconnecting edges $u_1u_2, u_2u_3, \ldots, u_{k-1}u_k$. We will often write the path $\pi = u_1 \rightsquigarrow u_k = u_1u_2 \oplus u_2u_3 \oplus \ldots \oplus u_{k-1}u_k$, where $\oplus$ denotes concatenation. A path that does not cross between the interior and the exterior of P or Q we call an *obstacle avoiding path.*

Consider two paths π_1 and π_2 that share a common vertex v. We say that v is a *vertex intersection* if the two paths cross each other at v. If they touch at v but do not cross, we call v a *touch point.* If some edges $e_1 \in \pi_1$ and $e_2 \in \pi_2$ intersect, we say that π_1 and π_2 have an edge intersection. Two paths intersect if they have either a vertex or an edge intersection.

If $\pi = u_1 \rightsquigarrow u_k$ is a path and $u_1 = u_k$, then the path is called a *tour.* A *simple* tour T has no duplicate vertices except the necessary first and last. A tour T is *weakly-simple* if it has no intersections except possibly for backtracking: $\ldots \oplus vv' \oplus v'v \oplus \ldots$. Thus, both simple and weakly-simple tours have well-defined interiors. Finally, in the context of the problem addressed in this paper, a weakly-simple tour with no P vertices is said to be *degenerate*.

A tour through all vertices of P and Q involves two types of edges: *polygonal edges* connecting consecutive P vertices or consecutive Q vertices and *cross-over edges* connecting P vertices with Q vertices or non-consecutive P or Q vertices. We will see later that non-consecutive P or Q vertices can not be adjacent in a shortest tour. We define $\parallel q_i \rightsquigarrow q_j \parallel$ to be the length of the polygonal path $q_i \rightsquigarrow q_j = q_iq_{i+1} \oplus \ldots \oplus q_{j-1}q_j$. Note that the path $q_i \rightsquigarrow q_j$ is clockwise, and that $(q_i \rightsquigarrow q_j) \oplus (q_j \rightsquigarrow q_i) = Q$. We similarly define $p_i \rightsquigarrow p_j$ and its length $\parallel p_i \rightsquigarrow p_j \parallel$.

Consider a polygon edge $p_kp_k^+$. We define a clockwise *detour*, cf. Figure 2.1, $d_{i,j}^k$ of $p_kp_k^+$, for any pair of not necessarily distinct Q vertices q_i and q_j to be the path $d_{i,j}^k = \mathcal{P}(p_k, q_i) \oplus (q_i \rightsquigarrow q_j) \oplus \mathcal{P}(q_j, p_k^+)$, where $\mathcal{P}(u, v)$ denotes a shortest obstacle-avoiding path from u to v.

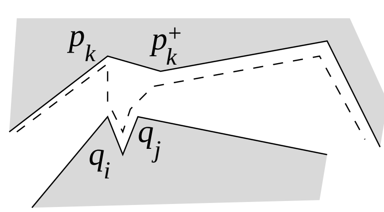

Figure 2.1: The detour $d_{i,j}^k$ from p_k to p_k^+ through $q_i \rightsquigarrow q_j$.

Note that the path $\mathcal{P}(u,v)$ may traverse points of P and Q and even result, when added to $q_i \rightsquigarrow q_j$ or to the tour sections before p_k or after p_k^+, in retracing vertices, cf. Figure 3.5. Two detours are *disjoint* if the sets of their P and Q vertices are disjoint. The *incremental cost* $c_{i,j}^k$ of the detour $d_{i,j}^k$ is

$$c_{i,j}^k = \| \mathcal{P}(p_k, q_i) \| + \| q_i \rightsquigarrow q_j \| + \| \mathcal{P}(q_j, p_k^+) \| - \| p_k \rightsquigarrow p_k^+ \| .$$

Let $d_{i,j}$ denote a cheapest clockwise detour through $q_i \rightsquigarrow q_j$ taken over all polygon sections $p_k p_{k'}$. That is

$$d_{i,j} = \arg\min_{p_k} \{c_{i,j}^k\} \qquad \text{and} \qquad c_{i,j} = \min_{p_k} \{c_{i,j}^k\}$$

Let A be a polygon and denote by Convex(A) its convex hull. Let a_i and a_j be two consecutive points on Convex(A). We call the points $a_i, a_{i+1}, \ldots, a_j$ together with the segments connecting them a *pocket* of A. We say that the segment $a_i a_j$ of the convex hull *crosses the pocket*. Cf. Figure 2.2.

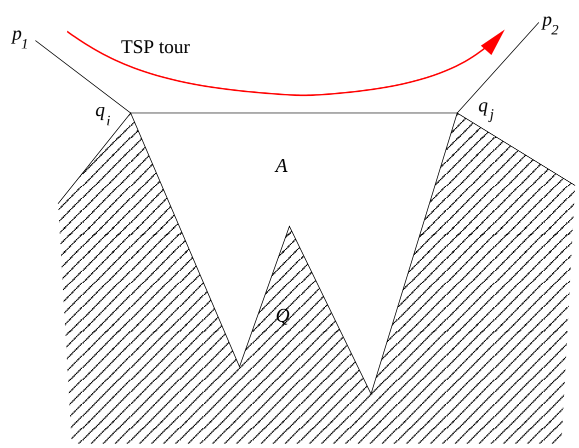

Figure 2.2: A TSP tour crosses a pocket A of Convex(Q).

3. Arbitrary Non-convex Obstacles

3.1 Structural Properties

In this section we provide a structural characterization of a shortest tour T through the vertices of P and Q. It will facilitate a transformation of the original problem to m shortest paths problems in an appropriately defined digraph.

We often implicitly use the following seemingly intuitive result:

Proposition 1. *Let abc be a triangle and C a convex chain in its interior between a and c of length $\| C \|$. Then $d(a,b) + d(b,c) > \| C \|$.*

Proof. The proof is by repeated application of the triangle inequality. Suppose x is some point in the interior of abc such that C is in the interior of the triangle axc and such that either ax or xc is tangent to a segment of the chain C. Clearly showing that $d(a,x) + d(x,c) < d(a,b) + d(b,c)$

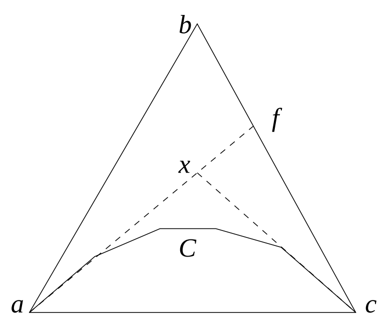

Figure 3.1: The construction from Proposition 1.

is sufficient to prove the proposition. Extend the line segment ax until it bisects edge bc and call the point of intersection f (Figure 3.1). Then we have that $ab + bc > af + fc > ax + xc$. (If C has more than two segments, it is always possible to choose x such that at least one segment of C is on ax or xc, and so we may iterate the procedure above by transforming abc and C into $a'xc'$

and C' where $C' \subset C$ connects a' to c' and has fewer segments than C.) Repeating this procedure we remove at least one segment of the chain at each step. The procedure therefore eventually terminates in a comparison of the triangle abc with a two segment (triangular) chain. $\square$

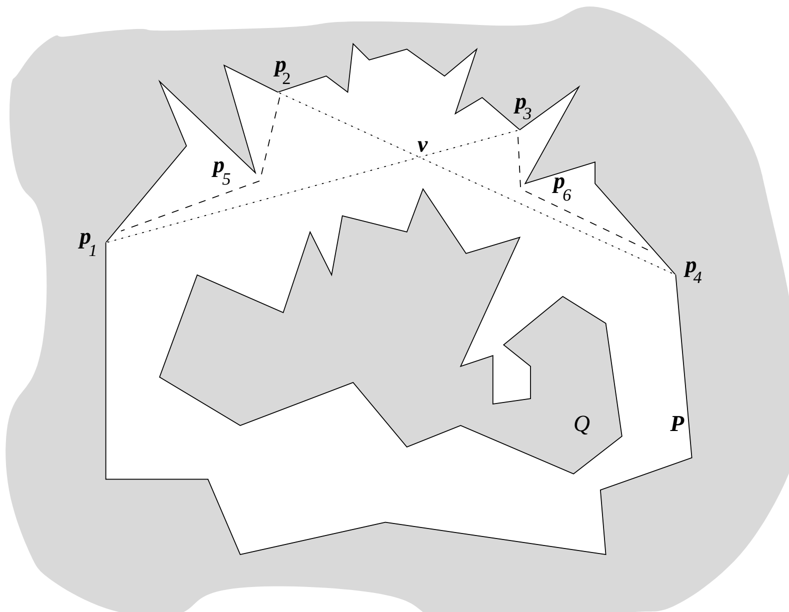

Figure 3.2: Lemma 2: Replacing p_1p_3 and p_2p_4 with p_1vp_2 and p_3vp_4 results in a tour of the same length. Relaxing to $p_1p_5p_2$ and $p_3p_6p_4$ results in a shorter tour.

Lemma 2. *An optimal tour T has intersections only on P and Q vertices.*

Proof. Assume to the contrary that the optimal tour T contains an intersection in the region between P and Q. We may consider the following four cases. Note that the vertices are necessarily distinct, since otherwise we would merely be describing backtracking.

1. **All four vertices are on P.** Call the four points p_1, p_2, p_3, and p_4 and call their point of intersection v. Suppose the shortest tour contains segments p_1p_3 and p_2p_4, as shown in Figure 3.2.

Then replacing p_1p_3 and p_2p_4 with either p_1vp_2 and p_3vp_4 or with p_1vp_4 and p_2vp_3 results in a tour of the same length, one of which must still be connected. Relaxing the tour away from v to the convex hull of the points in the triangles (in the diagram to p_5 and p_6) results, by Proposition 1, in a tour no longer than the original.

2. **Three vertices are on P and one on Q.** Let p_1, p_2 and p_3 denote the three vertices on P and q the vertex on Q such that v is the point of intersection of p_1p_3 and p_2q, as in Figure 3.3. As above, the intersecting edges extended to lines divide the plane into four regions, only two of which may contain vertices of Q.

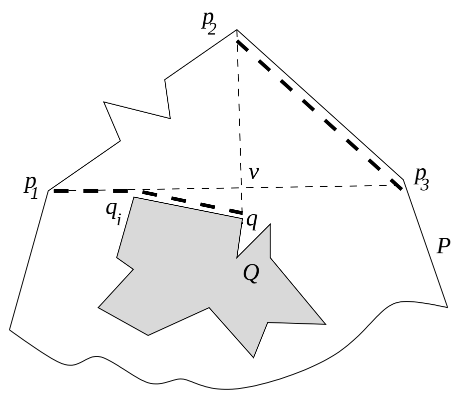

Figure 3.3: Lemma 2, case 2: three vertices on P, using a detour across Q. The bold dashed line shows a shorter tour section than the original non-bold dashed tour section.

We may substitute either p_1vq and p_2vp_3 or p_1vp_2 and qvp_3 without increasing the length of the tour. Relaxing the tour away from v to the convex hull of the points in the triangle again results in a tour of no greater length.

3. **Two vertices are on P and two on Q.** Let p_1, p_2, q_1, and q_2 denote the four vertices and suppose p_1q_2 and p_2q_1 intersect at a point v. We may substitute either p_1vq_1 and p_2vq_2 or p_1vp_2 and q_1vq_2 without increasing the length of the tour. There exist, therefore, paths $p_1 \rightsquigarrow p_2$ and $q_1 \rightsquigarrow q_2$ or else $p_1 \rightsquigarrow q_1$ and $p_2 \rightsquigarrow q_2$ that are shorter by Proposition 1.

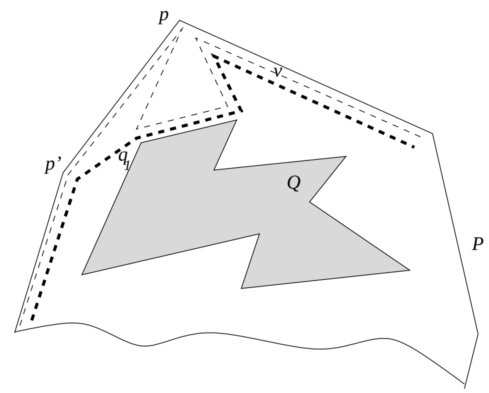

Figure 3.4: Lemma 2, case 3: one vertex on P. The bold dashed line shows a shorter tour section than the original non-bold dashed tour section.

4. **One vertex is on P and three are on Q.** By symmetry, this case is equivalent to Case 2.

$\square$

It would appear that for an optimal solution to the problem of finding a tour through the vertices of polygons P and Q the tour must visit each vertex exactly once. In fact, the tour may backtrack as it avoids obstacles, and so a shortest tour may visit vertices more than once, as Figure 3.5 illustrates.

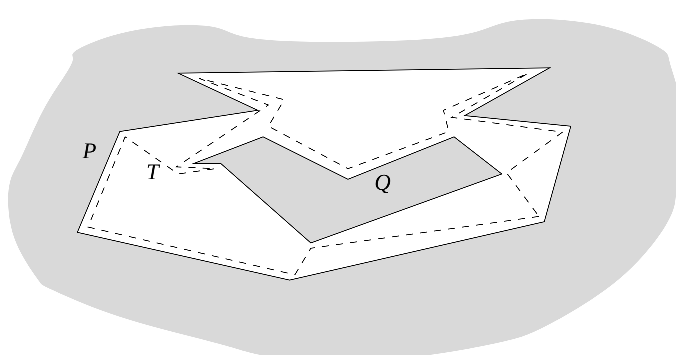

Figure 3.5: A shortest tour might visit some P and Q vertices more than once.

As we shall see, it would be very convenient to characterize shortest tours as sets of detours to Q from a cyclic traversal of P. Figure 3.5 illustrates that this is not possible. Nonetheless, we can so characterize shortest tours if we are willing to omit those vertices visited as part of detours.

Corollary 3. *There exists a shortest tour T on which P vertices preserve their cyclic order after all sub-tours of the form $\ldots pp'p \ldots$ are removed.*

That is, if we remove backtracking on P, the order of the remaining vertices on P is strictly cyclic.

Proof. Assume the contrary. Then the tour must intersect itself in the space between P and Q, which is forbidden by Lemma 2. $\square$

Detours in which the orientation on Q differs from the orientation on P are special:

Lemma 4. *If a shortest path, ignoring backtracking, is clockwise on P but counterclockwise on Q, then T has precisely one detour from P to Q.*

Proof. By the Jordan curve theorem [22] and Lemma 3, the detour on Q either covers all of Q or else it leaves vertices on Q which can not be reached by T without crossing the given detour. Since T visits all vertices of P and Q, the single detour covers all of Q. $\square$

One is tempted to use the results of [1] on the convex hulls of P and Q and then to "fix" the pockets: this approach fails. When P is convex, reads the temptation, we can partially compensate for the concavity of Q by computing the shortest tour on $P \cup \mathrm{Convex}(Q)$ and repairing the pocket crossings.

Lemma 5. *Let P be a convex polygon and Q a simple polygon contained within P. Let T be a shortest tour on $P \cup$ Convex(Q). Suppose that some segment of T crosses a pocket $B \subset Q$. Then there exists a shortest tour T' on $P \cup Q$ that follows the points of B.*

Proof. Suppose without loss of generality that the pocket B and the tour T are labeled as in Figure 3.6. The structural lemmas of [1] tell us that p_1 and p_2 must be adjacent on P. Suppose that some tour $\mathcal{T}$ on $P \cup Q$ is shorter than any tour that follows $p_1 q_i q_{i+1} \ldots q_j p_2$. Suppose that $\mathcal{T}$

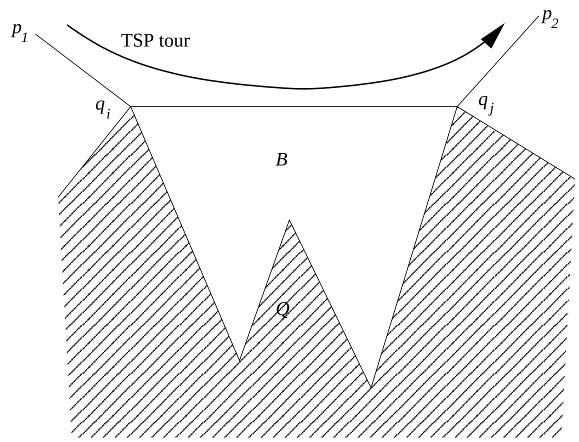

Figure 3.6: In the special case that P is convex, the case that the optimal tour of $P \cup \text{Convex}(Q)$ crosses a pocket.

enters B at some point q_k other than q_i. Then $q_i \rightsquigarrow q_{k-1}$ must return to P, which is a contradiction by the triangle inequality. $\square$

Unfortunately, it is not clear how to derive the related structural lemma we would need to extend our results directly by fixing each pocket. That is, suppose T is a shortest tour on $P \cup \text{Convex}(Q)$ and that some segment of T returns to P rather than cross a pocket $B \subset Q$ (Figure 3.7). Consider, for example, the problem of fixing B in the instance shown in Figure 3.8. We'll see in Section 3.2 that we can proceed in the absence of a repair strategy. Because of this, we do not consider the special case of convex P further.

3.2 Algorithm

In practice we can be clever by computing the convex hull of Q and noting that pockets shield the points within them. That is, suppose that B is a pocket of Q and I is the set of vertices inside (meaning not protruding from) B. Then if p is a point outside B that is not visible to either endpoint of B, then no point of I is visible to p. Nonetheless, this does not change the asymptotic behavior of our algorithm, for in the worst case (cf. Figure 3.8), $O(m)$ vertices of Q may be visible to $O(n)$ points of P.

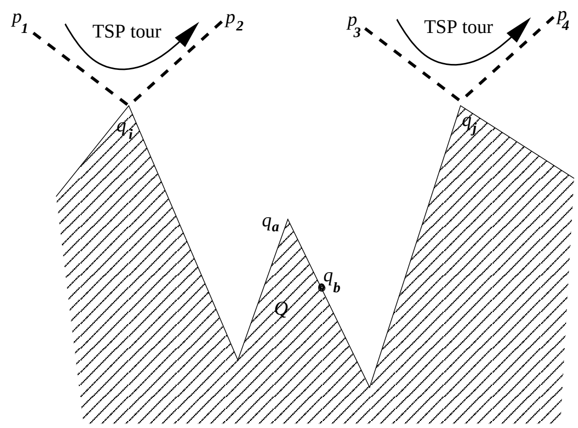

Figure 3.7: In the special case that P is convex, the case that the optimal tour of $P \cup \text{Convex}(Q)$ does not cross some pocket is no easier than the general case of two arbitrary polygons.

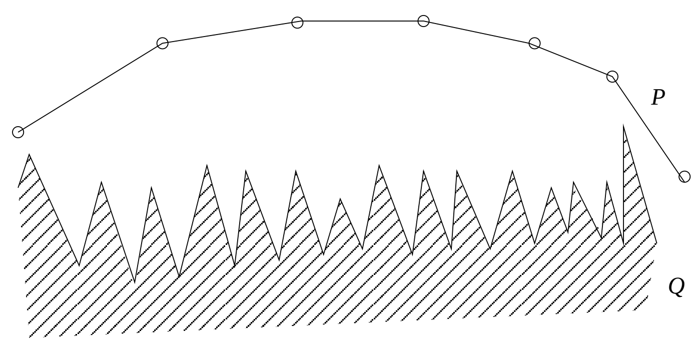

Figure 3.8: A pocket with $O(m)$ peaks and $O(n)$ vertices in P above..

By Lemma 4 we must consider two cases: the shortest tour with same orientation on P and Q and the shortest tour with opposite orientations. Suppose first that the tour has the same orientation on P and Q.

Let $G = (V, E)$ be a directed weighted graph with $V = \{x_0, \ldots, x_{m-1}, y_0, \ldots, y_{m-1}\}$ conceptually equal to the vertices of Q repeated twice, with q_i represented at x_i and again at y_i. The edges (v_i, v_j) have weight $w(v_i, v_j)$, where $w(x_i, y_j) = c_{i,j}$ and $w(y_i, x_j) = 0$ if $j = i$ or $j = i + 1$ (and ∞ otherwise). As noted earlier, subscripts are interpretted modulo m and $c_{i,j}$ is the cost of a cheapest detour $d_{i,j}$, as defined in Section 2. Then one of the m shortest paths from x_i to y_{i-1} corresponds to a shortest obstacle avoiding tour of P and Q:

Theorem 6. *For each $1 \leq h \leq m$ let π_h be the shortest path from x_h to y_{h-1}. Let π be the π_h with shortest length. Then π corresponds to a shortest same-orientation obstacle avoiding tour through all P and Q vertices.*

Proof. By Lemma 2, the path T corresponding to π is a weakly-simple tour and may be thought of as a cyclic tour of P plus the cost of one or more detours to Q. The detour $d_{i,j}$ is represented in the graph G as edges (x_i, y_j). The continuation along P until the next detour is represented by one of the zero weight edges from y_j to x_j or x_{j+1}. If a tour of shorter length on the vertices of P and Q existed, it would have a representation as a shortest path between some $x_{h'}$ and $y_{h'-1}$ in G. By construction, therefore, T is the shortest same-orientation tour as claimed. It has length $|P| + |\pi_h|$. □

Suppose, on the other hand, that the tour on Q has opposite orientation than on P. By Lemma 4, a shortest tour T will omit precisely one edge of P and one edge of Q. We need, therefore, only consider the shortest detour for each candidate edge of Q. The shortest tour, then, is the shorter of the shortest same-orientation tour and the shortest opposite-orientation tour:

Corollary 7. *Let P and Q be simple polygons as above. Then the shortest obstacle-avoiding tour of the vertices of P and Q is found by taking the shorter of the shortest tour obtained from Theorem 6 and the shortest opposite-orientation tour.*

Note that the principle difference (and increase in complexity) relative to the techniques in [1] relates to the increased complexity of computing the visibility graph.

3.3 Complexity

The algorithm in Section 3.2 leads us down two paths: finding the shortest same-orientation tour and finding the shortest opposite orientation tour. In the former case, we first find the least cost detours for each pair of Q vertices, then we find m shortest paths in a graph on $2m$ vertices. In the latter case we find and compare m shortest detours.

Theorem 8. *Finding the shortest tour of P and Q vertices requires $O(n^2m + nm^2 + m^2 \log m)$ time and $O(n^2 + m^2)$ space.*

Proof. The cost of computing the set of visible points for each $p \in P$ is $O(m(n+m))$ [10], and so $O(nm(n+m))$ for all p. Computing the points of P visible to vertices of Q is trivial, on the other hand, once we have computed Q's visibility from P, since the visibility graph is undirected. Computing visibility thus costs $O(nm(n+m))$. The space requirements are $O((n+m)^2)$ to store the edges of the graph.

Since we must consider $\Theta(nm^2)$ detours, the total cost of computing detours is $\Theta(nm^2)$. We use $O(m^2)$ space to store the values, since we minimize $c_{i,j}^k$ over k and so only store the $c_{i,j}$ and $d_{i,j}$.

Finding the shortest same-orientation detours requires $O(m^2 \lg m)$ time and $O(m^2)$ space to find the m shortest paths [19]. (Note that Karger et al.'s $m^* = O(m)$ in our application.) The total complexity is thus $O(m^2 \lg m + m^2 n)$ time and $O(nm + m^2)$ space.

The total complexity is thus $O(nm^2 + m^2 \log m + nm(n+m)) = O(n^2m + nm^2 + m^2 \log m)$ time and $O((n+m)^2 + m^2) = O(n^2 + m^2)$ space for the same-orientation shortest-tour.

To find the shortest opposite-orientation tour we compute the visibility graph and then compare m detours to Q for each of n adjacent vertices of P, using $O(nm)$ time and constant space (since we only need store the shortest). The total cost is thus $O(nm(n+m) + m^2 \log m) = O(n^2m + nm^2 + m^2 \log m)$ time and $O((n+m)^2) = O(n^2 + m^2)$ space. □

4. Convex Obstacles and Other Shortcuts

If Q is, in fact, nested inside P and if both polygons are convex [1], we may compute visibility by calipers [29]. The resulting complexity is then $O(m^2 \lg m + m^2 n)$ time and $O(nm + m^2)$ space, where m and n are defined as in this paper.

Theorem 9. *Let P and Q be convex polygons, Q completely in the interior of P. Finding the shortest tour of P and Q vertices that does not cross Q requires $O(m^2 \lg m + m^2 n)$ time and $O(nm + m^2)$ space.*

Proof. Computing the visibility graph of the vertices of P and Q takes time and space $O(mn)$, since we must consider and store mn pairs while the supporting tangents can be computed in amortized constant time using calipers due to the convexity of P and Q [29].

Since we must consider $\Theta(nm^2)$ detours, the total cost of computing detours is $\Theta(nm^2)$. We use $O(m^2)$ space to store the values, since we minimize $c_{i,j}^k$ over k and so only store the $c_{i,j}$ and $d_{i,j}$.

Finding the shortest same-orientation detours requires $O(m^2 \lg m)$ time and $O(m^2)$ space to find the m shortest paths [19]. The total complexity is thus $O(m^2 \lg m + m^2 n)$ time and $O(nm + m^2)$ space.

Finding the shortest opposite-orientation tour, from the above, requires $O(nm + m^2 n)$ time and $O(mn + m^2)$ space, since we must still compute the visibility graph and the $d_{i,j}$'s.

To find the shortest tour, we must find each of the above, and the result follows. $\square$

If Q is nested inside P, P convex and Q flattened to a line, Deineko et al. [8] showed an $O(mn)$ time and $O(n)$ space algorithm for computing a least cost tour, where n is the total number of vertices of which m lie on the line Q.

5. Non-planar Tours

A reasonable follow-on question regards extension to higher dimension. The answer, if not the proof, is short.

Theorem 10. *Let P and Q be non-intersecting polytopes in a three dimensional vector space X with additive distance function that obeys the triangle inequality. If P≠NP, computing a shortest TSP tour on the vertices of P and Q while considering the polytopes as obstacles is NP hard.*

Proof. We reduce to the planar case. Let G be a complete graph in the $z = 0$ plane (X with the z dimension projected out and under the same distance function restricted to the plane). Throughout this proof we will assume edges have weight equal to their length in X.

Choose $d_{\min}$ such that

$$0 < d_{\min} \ll \min\left(\min_{e \in E} w(e),\ \min_{\substack{e_i, e_j \in E \\ e_i \neq e_j}} |w(e_i) - w(e_j)|\right)$$

and choose $d_{\max} > |E| \max_{e \in E} w(e)$. Choose ϵ such that $d_{\min}/|E| \gg \epsilon > 0$.

We will now construct (cf. Figure 5.1) a polytope P from G and a second polytope Q whose only purpose is to fit a second polytope to the problem without changing the shortest tour on the vertices of P. We will construct the polytope P such that finding a shortest tour on P and Q will find, as a subproblem, a shortest tour on G. With this reduction we will have proved that the problem is NP hard.

We need to consider two essentially equivalent reductions: the case that Q is nested inside P and the non-nested case. Consider first the non-nested case. We begin with the graph G. We will construct a graph P that will define a polytope, which, by abuse of notation, we will also call P.

Start with $P = G$. We will add vertices (and so edges) to form our final polytope.

Choose some vertex $b \in V$ such that b is on the convex hull of V. Replace b by two distinct vertices, b_1 and b_2, in the $z = 0$ plane such that $|b_1 - b_2| < \epsilon$, $b_1 = b$, and b_2 is on the convex hull of V. We will call this augmented graph $G' = G - \{b\} + \{b_1, b_2\}$. By our definition of ϵ and

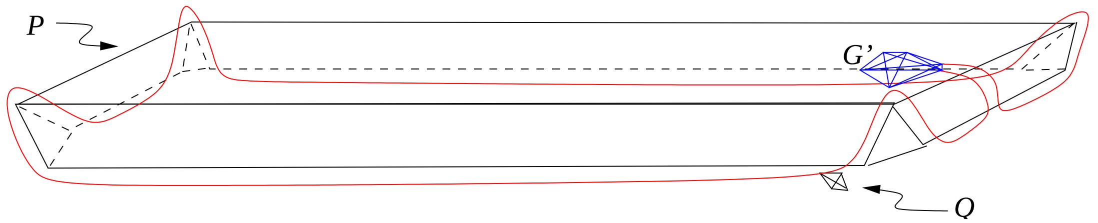

Figure 5.1: The construction of non-nested 3-D polytopes P and Q in oblique and side views. The traveling salesman tour (red) is free to visit any vertex of P and Q in any order as long as the vertices are visible to each other. By construction, the tour visits the vertices of G (blue) separately from the non-G vertices.

straightforward application of the triangle inequality, we have the following:

Claim 11. *If T is a shortest tour on G, then an identical tour T' on the augmented graph G' is a shortest tour on G'.*

We continue with the construction. In the $z = 0$ plane add two points c_1 and c_2 such that $b_1b_2c_2c_1$ is a square and c_1 and c_2 are outside the convex hull of V. Then add four additional points, the vertices of square C with sides of length $4d_{\max}$ such that c_1c_2 sits in the middle of one of the sides of the square and G lies entirely in the interior of C. Finally, consider the square D with truncated corners in the $z = -\epsilon$ plane with edge length $4d_{\max} - 2\epsilon$ centered under C. Add $4|E|$ points to P distributed uniformly on the interiors of the edges of D. The result, Figure 5.1, is a square pad with truncated bottom corners and with G embedded in its top face.

For the polytope Q, create a regular tetrahedron with edge lengths $\epsilon/2|E|$, positioned outside P a distance $\epsilon/2|E|$ from one of the sides of D, as indicated in Figure 5.1.

Claim 12. *A shortest tour T of $P \cup Q$ consists of a sub-tour T_1 of the vertices of G and a sub-tour T_2 of the vertices of $C \cup D \cup Q = (P \cup Q) - G$. Moreover, the transition between the two sub-tours T_1 and T_2 occurs on the line segments b_1c_1 and b_2c_2.*

Proof. Let T be a shortest tour on $P \cup Q$. By construction T will traverse the points of D in cyclic order with one or more detours to points of C and precisely one detour to points of Q. By construction, furthermore, the single detour to Q must occur only from the nearest vertices of D to Q. Moreover, any detour from $C \cup D$ to G other than at the $b_1c_1c_2b_2$ square would increase the cost of the tour by more than any savings from the change in order on $V(G)$, since a shortest tour on G has length less than $d_{\max}$. $\square$

With this characterization of shortest tours on $P \cup Q$ and Claim 11, it follows that a shortest tour on $P \cup Q$ computes as a sub-tour a shortest tour on G, which can not be done in polynomial time if P$\neq$NP. Therefore T can not be found in polynomial time unless P=NP.

The nested reduction is the same as the non-nested reduction except that we place Q inside P. $\square$

Note that no dimension higher than 3 can have a polynomial time solution, since the higher dimensions could be made thin enough that a solution in $d > 3$ dimensions could be used to find a solution in three dimensions.

6. Application: Navigation and Path Planning

A thesis devoted to a theoretical problem would be remiss not to find an application, however contrived, to justify the mathematics in the eyes of those for whom the mathematics itself is not adequately motivating. This chapter fills that niche.

The TSP arises naturally in the context of robot navigation or path planning, whether the robot is a mobile device or an arm fixed at one end. Consider a mobile robot presented with an environment filled with two contiguous (that is, polygonal) obstacles. Clearly the algorithm presented in this thesis is applicable to such an environment.

Consider Figure 6.1, the simplest non-pathological arrangement of two polygons: a rotated triangle embedded in another triangle. This example is sufficiently simple to visualize the graph G in an understandable way, Figure 6.2. Each edge of that graph represents a possible detour between vertices of the inside and outside triangles in Figure 6.1. Computing the shortest path in Figure 6.2 finds the shortest tour, illustrated in Figure 6.3.

We can also use our algorithm to compute the optimal tour of any closed channel without loops, as in Figure 6.4. In Figure 6.5 we add an extension loop to the channel, with the property that there is essentially a single way shortest way to traverse the extension. The resulting figure becomes two nested polygons. Since the tour of the connecting passage is trivial and doesn't affect the one-way tour of the channel, the result is an optimal tour of the channel that visits every vertex

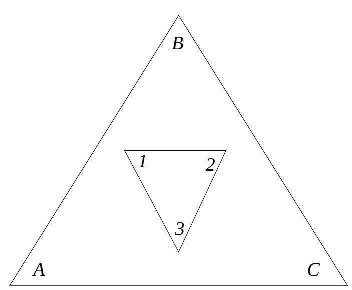

Figure 6.1: A triangular obstacle inside another triangle.

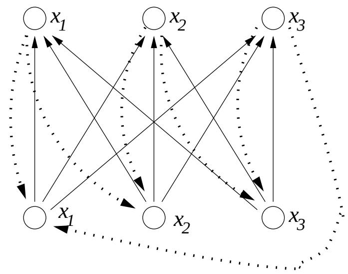

Figure 6.2: The shortest path graph from a triangular obstacle inside another triangle. The solid lines indicate the $c_{i,j}$, the dotted lines indicate the zero weight return edges.

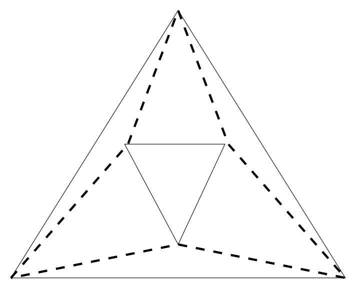

Figure 6.3: The shortest path visiting the vertices of two triangles.

along its length. Removing the connecting loop 6.6, we have a shortest tour of the channel.

As a final example, consider patrolling the exterior of a building as well the perimeter of its property, Figure 6.7. The robot is to visit every corner along the property line as well as every corner of the building. (Points along the edges could easily be designated corners if we wanted to guarantee guard coverage at certain intervals.) The resulting TSP tour is then a shortest guard tour that visits each point determined to be important.

Note that this problem is different from the Art Gallery Problem [23] and the Wandering Guard Problem [28], both of which concern points visible to the guard.

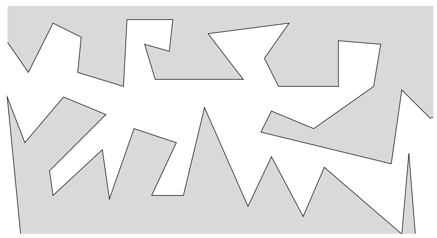

Figure 6.4: An irregular corridor.

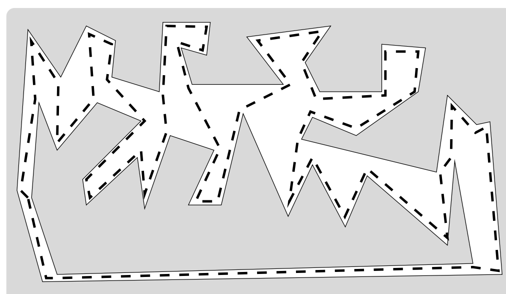

Figure 6.5: The shortest tour along the polygonal channel formed by an irregular corridor. Note that we form the channel by wrapping the ends around so that they form two polygons.

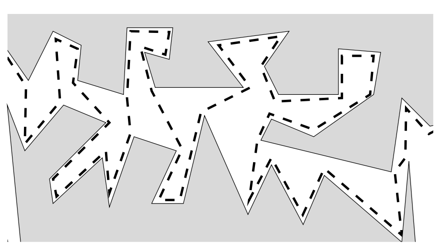

Figure 6.6: The shortest tour along an irregular corridor.

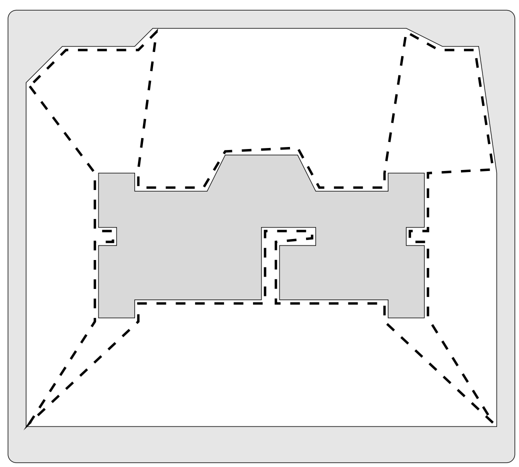

Figure 6.7: The shortest tour of watchpoints around a building (with service garage) and the perimeter of its property.

7. Conclusions

It would be nice to extend our results on the Euclidean TSP to the case of more than two polygons. The techniques used in this paper, unfortunately, are not readily extensible to more than $k = 2$ polygons. In particular, one might hope to place a third polygon around the existing two, but the technique of computing detours is not obviously adaptable to the third polygon.

If we could find an algorithm to treat cases $k > 2$, we can still make a few statements about those algorithms. First, the case $k > 3$ only makes sense if no more than one polygon encloses others, for the obstacle nature of the polygons that characterizes this version of the problem would make the problem insoluble if some polygon separated the set of polygons into two non-empty sets.

Second, the computational complexity must clearly increase without bound with k, for in the limiting case that each polygon approaches a point relative to the inter-polygon distances, this becomes the classic ETSP problem in its full generality.

Nonetheless, it may be possible, using techniques similar to those in this paper, to solve $k = 3$ in the special case that polygons Q_1 and Q_2 are completely enclosed within P and the convex hull of $Q_1 \cup Q_2$ does not intersect P. If one could do this, the result could probably be extended further by the same technique. As noted above, however, the reduction to ordinary ETSP would limit the usefulness of continuing in this manner.

To the best of our knowledge this is the first polynomial algorithm for this problem. The more general case of k polygons (for fixed $k \geq 2$) remains open. No lower bound for the problem is known beyond the trivial linear time needed to look at every vertex.

Bibliography

[1] Jeff Abrahamson, Ali Shokoufandeh, and Pawel Winter. Euclidean TSP between two nested convex obstacles. *Information Processing Letters*, 95:370–375, 2005.

[2] David Applegate, Robert Bixby, Vaŝek Chvàtal, and William Cook. On the solution of traveling salesman problems. *Documenta Mathematica*, Extra Volume ICM(III):645–656, 1998.

[3] G. Ausiello, P. Crescenzi, V. Kann, Marchetti-Spaccamela, Giorgio Gambosi, and Alberto M. Spaccamela. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer, New York, NY, 1999.

[4] Rainer E. Burkard, Vladimir G. Deineko, René van Dal, Jack A. A. van der Veen, and Gerhard J. Woeginger. Well-solvable special cases of the traveling salesman problem: a survey. *SIAM Review*, 40(3):496–546, sep 1998.

[5] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical Report 7, Carnegie-Mellon University, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1976. Cited on p. 37.

[6] M. Cutler. Efficient special case algorithms for the n-line planar traveling salesman problem. *Networks*, 10:183–195, 1980.

[7] V. G. Deineko, R. Rudolf, J. A. A. van der Veen, and G. J. Woeginger. Three easy special cases of the euclidean traveling salesman problem. Technical Report 17, Technische Univ., Graz, Austria, 1995.

[8] V. G. Deineko, R. van Dal, and G. Rote. The convex-hull-and-line traveling salesman problem: a solvable case. *Information Processing Letters*, 51:141–148, 1994.

[9] H. Edelsbrunner, G. Rote, and E. Welzl. Testing the necklace condition for shortest tours and optimal factors in the plane. In T. Ottmann, editor, *Automata, Languages, and Programming (Proceedings 14th Int. Coll. on Automata, Languages, and Programming (ICALP)*, number 266 in Lecture Notes on Computer Science, pages 364–375, Berlin, jul 1987. Springer Verlag.

[10] M. de-Berg et al. *Computational Geometry: Algorithms and Applications*. Springer, Berlin, 1998.

[11] Alfredo Garcia, Pedro Jordrâ, and Javier Tejel. A note on the traveling repairman problem. *Networks*, 40(1):27–31, 2002.

[12] M. R. Garey and D. S. Johnson. *Computers and intractability: a guide to the thory of NP-completeness*. Freeman, San Francisco, 1979.

[13] Gregory Gutin and Abraham P. Punnen. *Approximation Algorithms for Geometric TSP*, chapter 5. In [15], 2002. This is a cross-referencing INBOOK entry.

[14] Gregory Gutin and Abraham P. Punnen. *Polyhedral theory and branch-and-cut algorithms for the symmetric TSP*, chapter 2. In [15], 2002.

[15] Gregory Gutin and Abraham P. Punnen. *The traveling salesman problem and its variations*. Kluwer Academic Puublishers, Dordrecht, 2002.

[16] Refael Hassin and Shlomi Rubinstein. Better approximations for max TSP. *Information Processing Letters*, 75:181–186, 2000.

[17] D. S. Johnson and C. H. Papadimitriou. *Computational Complexity*, chapter 3, pages 37–85. In [20], 1985.

[18] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into Hilbert space. *Eur. J. Oper. Res.*, 19:68–75, 1985.

[19] D. R. Karger, D. Koller, and S. J. Phillips. Finding the hidden path: time bounds for all-pairs shortest paths. *SIAM Journal of Computing*, 22:1199–1217, 1993.

[20] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The Traveling Salesman Problem: A Guided Tour*. Wiley-Interscience, Chichester, 1985.

[21] Eugene Lawler. *Combinatorial Optimization: Networks and Matroids*. Dover, Mineola, New York, first edition, Jan 2001. This is an unabridged reprint of the 1976 book by the same name and author published by Holt, Rinehart & Winston.

[22] James Munkres. *Topology: A First Course*. Prentice-Hall, 1975.

[23] J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford International Series Of Monographs On Computer Science. Oxford University Press, New York, NY, USA, 1987.

[24] Christos H. Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4):765–768, 1981.

[25] J. C. Picard and M. Queyranne. The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, 26:86–110, 1978.

[26] G. Rote. *Two solvable cases of the Traveling Salesman Problem*. PhD dissertation, Technische Universität Graz, Institut für Mathematik, 1988.

[27] G. Rote. The n-line traveling salesman problem. *Networks*, 22:92–108, 1992.

[28] T. C. Shermer. Recent results in art galleries. In *Proceedings of the IEEE*, volume 80, pages 1384–1399, sep 1992.

[29] G. T. Toussaint. Solving geometric problems with the rotating calipers. In *Proceedings of IEEE MELECON*, Athens, Greece, may 1983.

[30] L. Trevisan. When Hamming meets Euclid: the approximability of geometric TSP and MST. In *Proc. 29th ACM Symp. Theory Comput.*, pages 21–39, 1979.

[31] V. Vazirani. *Approximation Algorithms*. Springer, Berlin, 2001.