Piecewise Linear Paths Among Convex Obstacles*

Mark de Berg

Vakgroep Informatica, Universiteit Utrecht,
Postbus 80.089, 3508 TB Utrecht, the Netherlands

Jiří Matoušek

Katedra aplikované matematiky, Universita Karlova,
Malostranské nám. 25, 118 00 Praha 1, Czech Republic and
Institut für Informatik, Freie Universität Berlin
Arnimallee 2-6, W 1000 Berlin 33, Germany

Otfried Schwarzkopf

Vakgroep Informatica, Universiteit Utrecht,
Postbus 80.089, 3508 TB Utrecht, the Netherlands

Abstract

Let $\mathcal{B}$ be a set of n arbitrary (possibly intersecting) convex obstacles in $\mathbb{R}^d$. It is shown that any two points which can be connected by a path avoiding the obstacles can also be connected by a path consisting of $O(n^{(d-1)\lfloor d/2+1\rfloor})$ segments. The bound cannot be improved below $\Omega(n^d)$; thus in $\mathbb{R}^3$, the answer is between n^3 and n^4. For open disjoint convex obstacles, a $\Theta(n)$ bound is proved. By a well-known reduction, the general case result also upper bounds the complexity for a translational motion of an arbitrary convex robot among convex obstacles. In the planar case, asymptotically tight bounds and efficient algorithms are given.

1 Introduction

The results presented in this paper are motivated by the following problem. Consider n disjoint convex obstacles of an arbitrary shape in the plane or in $\mathbb{R}^3$, and a convex robot R. Suppose that a position q of R can be reached from a position p by a translational motion of R avoiding the obstacles. If such a motion is piecewise linear, we can measure its complexity by the number of linear segments. Is it possible to bound the minimum necessary complexity of such a motion solely in terms of n, the number of obstacles? This does not seem obvious even if R is a single point. Note that the boundaries of two convex shapes in the plane may intersect an arbitrary number of times.

It turns out that the minimum necessary complexity of a motion can indeed be bounded in this way, in an arbitrary fixed dimension d.

By a well-known method we can reduce the problem of translational motion of a convex robot among convex obstacles to the case of point robot among convex obstacles (every obstacle B is replaced by $B - R$, the Minkowski difference of the obstacle and the robot).

*This research was supported by the Netherlands' Organization for Scientific Research (NWO) and partially by the ESPRIT III Basic Research Action 6546 (PROMotion). J. M. acknowledges support by Humboldt Research Fellowship. Part of this research was done while he visited Utrecht University.

Then, however, the obstacles are not necessarily disjoint anymore, even if the original ones are. This leads us to investigating the case of point moving among completely general (possibly intersecting) convex obstacles. Here we obtain an $\Omega(n^d)$ lower bound and an $O(n^{(d-1)\lfloor d/2+1\rfloor})$ upper bound. In the plane, this gives $\Theta(n^2)$ complexity. In $\mathbb{R}^3$, we get $\Omega(n^3)$ from below and $O(n^4)$ from above, and the gap gets larger in higher dimensions. We conjecture that the truth is closer to the lower bound.

The obstacles arising as Minkowski differences of disjoint convex bodies with the same robot are not completely arbitrary. For instance, in the plane they form a collection of *pseudodisks* (the boundaries of any two intersect in at most two points), and we can exploit this property and show a linear upper bound on the motion complexity. In higher dimensions we currently have no improvement over the general case, although we can only give an $\Omega(n^2)$ lower bound in $\mathbb{R}^3$.

We also consider the special case of a point robot moving among disjoint convex obstacles. In this case, we obtain a $\Theta(n)$ bound for the necessary number of segments[1] to connect two points. This result follows easily from the planar case.

So far we have concentrated on the combinatorial bounds. The next step is to obtain efficient algorithms for computing paths with a possibly small number of segments. Currently, we consider algorithms more systematically for the planar case only. To retain generality, we suppose that the convex obstacles are given by suitable *oracles* which can answer various queries (like "find a line separating obstacles B_i, B_j"). We obtain nearly linear-time algorithm for the pseudodisk case and a nearly $O(n^2)$ algorithm for the general intersecting case, provided that the oracles can be implemented efficiently. If the obstacles are convex polygons or they are bounded by algebraic curves of bounded degree (e.g., by circular arcs), the oracle calls can be implemented in logarithmic time. We also have an efficient implementation for obstacles which are Minkowski differences. This leads to the following result: if the obstacles are convex polygons with a total of m vertices and the robot is a convex k-gon, a translational motion with $O(n)$ segments can be found in $O(n \log^2(k+m))$ time (assuming the vertices of each polygon are stored in a clockwise order in an array).

We now review the related results and approaches in the literature. As far as we know, the problem for general convex obstacles in our sense has never been considered; we regard this new point of view as one of the main contributions of our work. The usual approach is to consider obstacles with piecewise linear boundaries and express the bounds in terms of the total combinatorial complexity. Our approach can bring a substantial improvement e.g., when the piecewise linear surface arises as an approximation of some curved convex shape.

The motion planning problem has received considerable attention; we mention only the works more or less directly related to our results, referring e.g., to [Lat91, SS90] for a wider background and literature.

A usual approach to the motion planning problem is to compute a full combinatorial description of the *free space* (the set of all admissible configurations of the robot). For the translational motion of a convex robot among convex obstacles in the plane, this approach was exploited by Kedem et al. [KLPS86]. If the robot is a convex k-gon and the environment consists of n convex polygons with a total of m edges, the complexity of this

[1]This number is usually called the *link distance* of p and q.

free space is $O(kn + m)$, and it can be computed in time $O(km \log^2(k + m))$.

Our approach is closer to another motion planning method, which tries to replace the whole environment by some "simpler" subset, typically of smaller dimension. This is sometimes referred to as the *retraction method*. For instance, in the plane, one can compute a generalized Voronoi diagram (a Voronoi diagram of the obstacle edges, where distances are measured with a distance that is determined by the shape of R). This Voronoi diagram captures the connectivity information about the free space, and can be used to perform motion planning. The complexity of this diagram is in $O(kn + m)$ as well, and it can be computed in time $O(km \log(k + m))$, see [CD85]. Recently, however, Sifrony [Sif89], has shown that the diagram can actually be stored in space $O(k+m)$ such that motion planning queries which ask for the existence of a path can be answered in time $O(\log m)$. The running time for the computation of the compacted diagram is still $O(mk \log n)$. Furthermore, Kao and Mount [KM91] have shown that a path can actually be computed in time $O(m \log m \log^2 k)$. Although they can describe it in space $O(k+m)$, the path computed by this method still has complexity $\Theta(km)$. If both k and m are large and m is significantly larger than n, our results thus provide a significant improvement.

An extension of the above quoted algorithms to the case of non-linear obstacle boundaries (circular arcs, say) appears nontrivial, while our approach only requires a relatively straightforward implementation of the oracle. Such non-linear obstacles are considered important in practical situations, see e.g. Alt and Yap [AY89].

Another elegant method of the retraction type is due to Canny and Donald [CD88]. Roughly speaking, for every pair of features of the obstacles, they define a "generalized bisector", and they look for the path in a subset of the environment defined by these bisectors. This resembles our methods, and their work also contains topological considerations with a similar flavor as our "expansion lemma" (Lemma 2). In some sense their method is more general, since it can also handle rotational motions etc. On the other hand, with our more restricted assumptions, we are able to get results for completely general convex obstacles, while the result of [CD88] is (necessarily) expressed in terms of features of the obstacles. Also, they only give the results for motion planning in 2 and 3 dimensions (with rotation and translation and with the robot and obstacles bounded by algebraic surfaces).

2 Preliminaries and notation

By $[n]$ we denote the set $\{1, 2, \ldots, n\}$, and $\binom{A}{k}$ denotes the set of all k-element subsets of a set A.

For a set $X \subseteq \mathbb{R}^d$, we let $\overline{X}$ denote the closure of X, ∂X the boundary of X and $\mathrm{int} X$ the interior of X.

We will mainly consider the path of a point-sized robot among a family $\mathcal{B} = \{B_1, \ldots, B_n\}$ of n convex obstacles. The endpoints of the path will be denoted by p and q.

Throughout the paper, we will consider the obstacles as *open* convex bodies. This means that the path is allowed to "glide" along the boundary of the obstacles and so the environment is closed.

In various topological considerations, it is convenient to have the environment also bounded and thus compact. Then we may assume that the obstacles are bounded as well.

In order to model also the case of unbounded obstacles (e.g., in practice some obstacle might be too large to go around it) we imagine that the obstacles are enclosed in a large enough closed cube U. We can regard U as being bounded by $2d$ special open convex obstacles ("walls") $W_1, \ldots, W_{2d}$, which can also be assumed bounded. Our original obstacles are then allowed to intersect the walls. In this way, we can assume that all the obstacles we deal with are bounded, and the relevant connected component E of their complement $\mathbb{R}^d \setminus \bigcup_{i=1}^n B_i$ is also bounded and thus compact. Note that we do not lose generality here, since in the unbounded case we can always take a large enough cube so that it contains a path with minimum complexity.

Let us now assume that $B_1, \ldots, B_n$ are open convex polytopes. In such a situation, there exists a shortest p-q path within E, and each shortest path γ is piecewise linear (since E can be triangulated into a finite number of simplices, and within every simplex we could replace a curved portion of γ by a segment).

Let γ be a shortest p-q path. We call a point $x \in \gamma$ where two segments of different directions meet a *turn* of γ.

Lemma 1 *Let γ be a shortest p-q path, x a turn of γ. Then x belongs to the closure of a $(d-2)$-dimensional face of some polytope B_i.*

Proof: Let the turn x be a common endpoint of the segments a, b of the path γ. Let π be the plane containing a and b, and let α be the angle in π defined by a, b with apex x (considered as a subset of π, not as a number). Since γ is a shortest path, arbitrarily small neighborhood of x in the angle α has to be intersected by some obstacle (otherwise we could take a shortcut on γ). Since we have finitely many obstacles, one obstacle B_i has to intersect an arbitrarily small neighborhood of x in α, so $x \in \partial B_i$. And since B_i is an open convex polytope, x cannot be an interior point of a facet of B_i. $\square$

Let us call a B_i as in the above lemma *supporting γ at x*.

3 Expanding convex bodies

In this section we develop a technique for dealing with the problem of paths among convex obstacles; it will be applied to obtain various specific bounds in the next sections. We enlarge the given obstacles in such a way that p and q remain connected by a path, and we ensure that the enlarged obstacles are convex polytopes with a suitably bounded combinatorial complexity. Then we find a path using the combinatorial features (edges, facets) of these polytopes.

The following lemma shows that we can safely expand the convex obstacles into larger ones as long as this does not create any new d-wise intersections between them.

Lemma 2 *Let $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$ and $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ be collections of bounded open convex bodies in $\mathbb{R}^d$, and p, q points not belonging to any B_i or C_i. Suppose that the following conditions are satisfied:*

(i) $B_i \subseteq C_i$, $i = 1, \ldots, n$

(ii) *For any set $I \in \binom{[n]}{d}$ with $\bigcap_{i \in I} B_i = \emptyset$ we have $\bigcap_{i \in I} C_i = \emptyset$.*

Then p and q can be connected by a path avoiding all B_i if and only if they can be connected by a path avoiding all C_i.

Here is an intuitive explanation why the lemma holds, at least in dimension 2 and 3. Imagine that we start with p and q connected by a rubber string avoiding the B_i's. Then, as we continuously expand the B_i's to the C_i's, we can continuously deform the path, keeping it outside the growing bodies. In the plane, a moment when this is not possible anymore may appear only when two previously disjoint bodies touch and cut our path. In 3-space, the critical situation is when three bodies form a narrow channel through which the path passes, and this channel closes as a triple intersection appears. It appears that this idea can be turned into a rigorous proof, but technically it seems easier (or shorter, at least) to use powerful ready-made tools from algebraic topology (most notably, Alexander's duality theorem) for another type of proof.

The intuition behind this proof is as follows. Suppose that there is no p-q path avoiding the C_i's. Then there must be a "reason" for this, something like a $d-1$ dimensional surface c within $\bigcup \mathcal{C}$ which also separates p and q. In the plane it must be a topological circle; in higher dimensions it can be a topological sphere or something more complicated, like a torus. Now this separating "surface" c can be triangulated finely enough, and then "pulled" into $\bigcup \mathcal{B}$ using the intersection condition, while it retains its separating property. Thus p and q are separated also by the bodies of $\mathcal{B}$.

The formal proof uses notions and results of algebraic topology (homology and cohomology theory), which can be found e.g. in [Mun84]. Its understanding is not needed for the rest of this paper.

Proof: The proof proceeds in two stages. First, we prove a similar lemma where, instead of open convex bodies, the B_i's and C_i's are compact polyhedra (which matches the assumptions of theorems we use). Then we derive the form with open bodies by a limit argument.

Lemma 3 *Let $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$ and $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ be collections of compact convex polyhedra in $\mathbb{R}^d$, satisfying conditions (i), (ii) of Lemma 2. Then p and q can be connected by a path avoiding all B_i if and only if they can be connected by a path avoiding all C_i.*

Proof: Set $X_{\mathcal{B}} = \bigcup \mathcal{B}$, $X_{\mathcal{C}} = \bigcup \mathcal{C}$. Suppose that points p, q belong to different components of $\mathbb{R}^d \setminus X_{\mathcal{C}}$; we want to show that they also belong to different components of $\mathbb{R}^d \setminus X_{\mathcal{B}}$. We consider the 0-dimensional reduced homology groups $\tilde{H}_0(\mathbb{R}^d \setminus X_{\mathcal{C}})$ and $\tilde{H}_0(\mathbb{R}^d \setminus X_{\mathcal{B}})$. We can choose the 0-cycle $p - q$ as one of the free generators of $\tilde{H}_0(X_{\mathcal{C}})$. We will show that there is a injective homomorphism mapping $\tilde{H}_0(\mathbb{R}^d \setminus X_{\mathcal{C}})$ into $\tilde{H}_0(\mathbb{R}^d \setminus X_{\mathcal{B}})$. This homomorphism will be natural; the relevant fact needed here is that is preserves the identity of the components. Such a homomorphism cannot map the 0-cycle $p - q$ to zero, and hence p and q lie in distinct components also in $\mathbb{R}^d \setminus X_{\mathcal{B}}$. (Notice that we cannot, in general, expect the homomorphism to be onto, since some connected components might have disappeared completely when the B_i's are expanded to the C_i's.)

Each B_i is a closed convex polyhedron bounded by a finite number of facet hyperplanes. If we triangulate the arrangement of all the facet hyperplanes, $\mathbb{R}^d$ becomes a simplicial complex[2], $X_{\mathcal{B}}$ is its subcomplex, and each B_i is a subcomplex of $X_{\mathcal{B}}$.

[2]This is not quite true, since the simplices should be compact, while one usually permits also "unbounded simplices" in the triangulation of unbounded cells of a hyperplane arrangement. To remedy this, we can either add a point ∞ at infinity and work with $\mathbb{R}^d \cup \{\infty\}$ as a topological sphere, or we can triangulate the unbounded cells by countably many ordinary simplices.

We can now apply (a particular case of) the Alexander's duality theorem (see [Mun84]), and we get the 0-dimensional reduced homology group of $\mathbb{R}^d \backslash X_{\mathcal{B}}$ is (naturally) isomorphic to the $(d-1)$-dimensional reduced cohomology group $\tilde{H}^{d-1}(X_{\mathcal{B}})$.

As mentioned above, the sets of $\mathcal{B}$ form a covering of $X_{\mathcal{B}}$ by its subcomplexes. We let $\mathcal{N}(\mathcal{B})$ denote the nerve of this covering. This is an abstract simplicial complex, with vertex set $[n]$ (the set indexing $\mathcal{B}$). A subset $\sigma \subseteq [n]$ forms a simplex in $\mathcal{N}(\mathcal{B})$ iff $\bigcap_{i\in\sigma} B_i \neq \emptyset$. Since the B_i's are convex, any set in the covering and also any intersection of these sets is convex and thus contractible. In such a situation, it is known that the homology groups of $X_{\mathcal{B}}$ and the homology groups of $\mathcal{N}(\mathcal{B})$ are isomorphic (Folkman-Leray theorem, see [Rot88]). By the universal coefficient theorem, also the cohomology groups are isomorphic, so in particular $\tilde{H}_0(\mathbb{R}^d \backslash X_{\mathcal{B}}) \cong \tilde{H}^{d-1}(\mathcal{N}(\mathcal{B}))$. A similar argument applies for $\mathcal{C}$, so it remains to provide a (natural) injective homomorphism of $\tilde{H}^{d-1}(\mathcal{N}(\mathcal{C}))$ into $\tilde{H}^{d-1}(\mathcal{N}(\mathcal{B}))$.

By condition (i) of the lemma, $\mathcal{N}(\mathcal{B})$ can be regarded as a subcomplex of $\mathcal{N}(\mathcal{C})$, and by (ii) the sets of $(d-1)$-dimensional simplices are identical in both complexes. The inclusion map $i : \mathcal{N}(\mathcal{B}) \to \mathcal{N}(\mathcal{C})$ induces a homomorphism of the $(d-1)$-dimensional cohomology groups $\tilde{H}^{d-1}(\mathcal{N}(\mathcal{C})) \to \tilde{H}^{d-1}(\mathcal{N}(\mathcal{B}))$, and it is easy to check that the fact that $\mathcal{N}(\mathcal{C})$ has no new $(d-1)$-dimensional simplices implies that this homomorphism is injective. This concludes the proof of Lemma 3. $\square$

It remains to derive Lemma 2 from the polyhedral version, Lemma 3. Before we do this, we prove a lemma showing how to enlarge the obstacles into polytopes of bounded complexity without creating new d-wise intersections. This lemma will also be useful later.

Lemma 4 *Let $B_1, \ldots, B_n$ be bounded open convex bodies in $\mathbb{R}^d$, and let p, q be points not belonging to any B_i. Then one can construct bounded open convex polyhedra $C_1, \ldots, C_n$ avoiding p, q, such that $B_i \subseteq C_i$, for any $I \in \binom{[n]}{d}$, $\bigcap_{i\in I} C_i = \emptyset$ iff $\bigcap_{i\in I} C_i = \emptyset$, and each C_i has $O(n^{d-1})$ facets.*

Proof: Let $\tilde{U}$ be a large enough open cube containing all B_i. We construct $C_1, \ldots, C_n$ inductively. Suppose that $C_1, \ldots, C_i$ have already been defined. Let $\mathcal{F}_i$ be a collection of convex bodies defined as follows. $\mathcal{F}_i$ contains p, q and each nonempty $(d-1)$-wise intersection of the form

$$F_{i_1,\ldots,i_{d-1}} = C_{i_1} \cap C_{i_2} \cap \ldots \cap C_{i_k} \cap B_{i_{k+1}} \cap B_{i_{k+2}} \cap \ldots \cap B_{i_{d-1}}$$

with $i_1 < i_2 < \ldots < i_k \leqslant i < i+2 \leqslant i_{k+1} < \ldots < i_{d-1}$, such that $B_{i+1} \cap F_{i_1,\ldots,i_{d-1}} \neq \emptyset$. For each $F \in \mathcal{F}_i$, we find an open halfspace h_F containing B_{i+1} and disjoint from F. We then let C_{i+1} be the intersection of all halfspaces h_F, $F \in \mathcal{F}_i$ and the cube $\tilde{U}$. It is straightforward to verify that the family $\{C_1, \ldots, C_n\}$ thus constructed satisfies the requirements of the lemma. $\square$

We are ready to finish the proof of Lemma 2. Let $\mathcal{B}, \mathcal{C}$ be the families as in Lemma 2. First we deal with the "smaller" bodies of $\mathcal{B}$. For any $I \in \binom{[n]}{d}$ choose a point $x_I \in \bigcap_{i\in I} B_i$, provided that this intersection is nonempty, and define

$$B_i' = \operatorname{conv}\{x_I;\ i \in I \in \binom{[n]}{d},\ \bigcap_{i\in I} B_i \neq \emptyset\}\,.$$

Each B_i' is a closed convex polyhedron contained in B_i, and it is easily seen that if the claim Lemma 2 fails for some $\mathcal{B} = \{B_1, \ldots, B_n\}$, then it also fails for $\mathcal{B}' = \{B_1', \ldots, B_n'\}$.

Let us now look at the C_i's. These are first expanded into bounded complexity open polyhedra C_i' using Lemma 4. Each of these is a bounded open convex polytope whose combinatorial complexity is bounded solely in terms of n and d. They have the same d-wise intersection pattern as the C_i's. Therefore it suffices to prove Lemma 2 for the families $\mathcal{B}'$ (compact polyhedra) and $\mathcal{C}' = \{C_1', \ldots, C_n'\}$.

We still cannot use Lemma 3 directly, since $\mathcal{C}'$ consists of *open* polyhedra; we will thus express every C_i' as a limit of closed polyhedra. The bounded complexity of the C_i' is important for arguing about limit of paths. For every $C_i' \in \mathcal{C}'$, choose a sequence $C_i^{(1)}, C_i^{(2)}, \ldots$ of closed convex polyhedra with the following properties:

- for all j, $B_i' \subseteq C_i^{(j)} \subseteq C_i'$
- $C_i' = \bigcup_{j=1}^{\infty} C_i^{(j)}$
- the number of facets of $C_i^{(j)}$ is no larger than the number of facets of C_i'.

It is straightforward to produce such a sequence, by shifting the facet hyperplanes of C_i inwards by sufficiently small amounts converging to zero. Set $\mathcal{C}_j = \{C_1^{(j)}, \ldots, C_n^{(j)}\}$. For every j, the families $\mathcal{B}'$ and $\mathcal{C}_j$ satisfy the assumptions of the "polyhedral" version—Lemma 3—so if there is a p-q path avoiding $\mathcal{B}'$, then there is also a path avoiding $\mathcal{C}_j$. It is easy to see, using the bounded combinatorial complexity of the polyhedra of $\mathcal{C}_j$, that there must also exist a piecewise linear p-q path avoiding $\mathcal{C}_j$ with the number of segments bounded by some number N *independent of* j. Let γ_j denote some such path.

All the paths γ_j are contained in the unit cube U, which is a compact set. Because every γ_j is determined by at most an N-tuple of points (the endpoints of the segments) in U, there exists a subsequence $\gamma_{j_1}, \gamma_{j_2}, \ldots$, which converges to a path γ (meaning that each γ_{j_i} has the same number of defining points as γ and the defining points of γ_{j_i} converge to the appropriate defining points of γ). It is straightforward to verify that this limit path γ avoids $\mathcal{C}'$. This finishes the proof of Lemma 2. ◻

4 General obstacles in higher dimension

In this section we consider arbitrarily intersecting convex obstacles.

Theorem 5 *Let $\mathcal{B}$ be a collection of n bounded convex bodies in $\mathbb{R}^d$. Then any two points in the same connected component of $\mathbb{R}^d \setminus \bigcup \mathcal{B}$ can be connected by a path consisting of $O(n^{(d-1)\lfloor d/2+1 \rfloor})$ segments.*

Proof: Using Lemma 4, we first replace $\mathcal{B}$ by a family $\mathcal{C}$ of bounded open convex polytopes, each with $O(n^{d-1})$ facets, without creating new d-wise intersections. Lemma 2 guarantees that there exists a p-q path avoiding the obstacles of $\mathcal{C}$. Hence, the theorem follows from Lemma 6 below. ◻

Lemma 6 *Let $\mathcal{C}$ be a collection of n bounded open convex polytopes in $\mathbb{R}^d$, each defined as the intersection of at most m halfspaces. Then any two points p, q in the same connected component of $\mathbb{R}^d \setminus \bigcup \mathcal{C}$ can be connected by a path consisting of $O(n^{d-1} m^{\lfloor d/2 \rfloor})$ segments.*

Proof: Consider a shortest p-q path γ. To each turn x of the path γ, we assign a certain point $v(x)$. Let $C_0 \in \mathcal{C}$ be a supporting obstacle for γ at x.

For dimension $d = 2$, $v(x)$ will be simply equal to x, which is a vertex C_0. For dimension $d = 3$, x is either a vertex of C_0, in which case we set $v(x) = x$, or it lies on an edge e of C_0. We go along this edge from x in one (arbitrarily chosen) direction, until we either reach its end vertex u (a vertex of C_0) or we reach a point v where e enters the interior of some other obstacle $C_1 \in \mathcal{C}$. In the first case, we set $v(x) = u$, in the second case $v(x) = v$. For a general dimension, the formal definition of $v(x)$ is somewhat complicated, and can be done inductively as follows.

We set $y_1 = x$, $\mathcal{D}_1 = \{C_0\}$ and $H_1 = \{h_1, h_2\}$, where h_1, h_2 are two facet hyperplanes of C_0 containing x (they exist by Lemma 1). These will be objects entering the first step of the construction. On the beginning of a general (ith) step, we will have a current point y_i, a current collection $\mathcal{D}_i \subseteq \mathcal{C}$ of at most i polytopes of $\mathcal{C}$ and a current set H_i of $i+1$ hyperplanes, each being a facet hyperplane of some $C \in \mathcal{D}_i$. The intersection $f_i = \bigcap H_i$ of the hyperplanes is a $(d-i-1)$-flat. The point y_i lies in f_i and on the boundary of the intersection $\bigcap \mathcal{D}_i$ of the polytopes of $\mathcal{D}_i$.

In the ith step, we choose a ray ρ_i originating at y_i and lying within the flat f_i. We consider the points of ρ_i linearly ordered along ρ_i (y_i is the first point). For every $C \in \mathcal{C}$, we define a point $v_C \in \rho_i$; we distinguish two cases.

(i) If $C \in \mathcal{D}_i$, we let v_C be the last point of ρ_i in the boundary ∂C. It is easy to check that as ρ_i is leaving the boundary at v_C, it has to cross some facet hyperplane of C at v_C. We denote this hyperplane by h_C.

(ii) If $C \notin \mathcal{D}_i$, we let v_C be the last point of ρ_i before ρ_i enters the (open) polytope C. If ρ_i does not intersect C, we put $v_C = \infty$. If v_C is not ∞, then again ρ_i crosses a facet hyperplane of C at v_C; we choose one such hyperplane and denote it by h_C.

We now let y_{i+1} be the first point among the v_C, $C \in \mathcal{C}$ (it may happen that $y_{i+1} = y_i$). We choose some C with $v_C = y_{i+1}$; if there are more possibilities, we prefer a $C \in \mathcal{D}_i$ (case (i)) over $C \notin \mathcal{D}_i$ (case (ii)). We define $H_{i+1} = H_i \cup \{h_C\}$ and $\mathcal{D}_{i+1} = \mathcal{D}_i \cup \{C\}$. After $d-2$ steps the construction finishes, with the flat $f_{d-1} = \bigcap H_{d-1}$ being 0-dimensional and containing only the point y_{d-1}, which we define to be $v(x)$.

We need to establish that the step preserves the invariant (the above listed properties of $y_i, H_i, \mathcal{D}_i$), which is straightforward. The newly added hyperplane h_C is not parallel to the ray ρ_i, and thus neither to f_i, hence $\dim f_{i+1} < \dim f_i$. The point y_i lies on the boundary of the intersection of the polytopes of $\mathcal{D}_i$, and as we go along the ray ρ_i, we can only leave this boundary by passing some point v_C for $C \in \mathcal{D}_i$. If y_{i+1} is defined as v_C for $C \in \mathcal{D}_i$ (case (i)), we have $\mathcal{D}_{i+1} = \mathcal{D}_i$ and so y_{i+1} lies in $\partial(\bigcap \mathcal{D}_i)$. If the polytope C defining y_{i+1} does not belong to $\mathcal{D}_i$, there is some portion of ρ_i after y_{i+1} which still lies in the boundary of $\bigcap \mathcal{D}_i$, but it also lies inside the new C. This implies that y_{i+1} is on the boundary of $\bigcap \mathcal{D}_{i+1} = C \cap \bigcap \mathcal{D}_i$ as well.

The above construction guarantees that x can be connected to $v(x)$ by the path $x = y_1, y_2, y_3, \ldots, y_{d-1} = v(x)$, which consists of at most $d-2$ segments and avoids the obstacles.

A portion of the path γ between two turns x, x' with $v(x) = v(x')$ can thus be replaced by a portion consisting of at most $2(d-2)$ segments.

We estimate the number of vertices which may appear as $v(x)$ for some x. By construction, any such point is a vertex of the intersection of some $(d-1)$-tuple of the obstacles. Such an intersection is defined by $O(m)$ halfspaces, thus it has $O(m^{\lfloor d/2 \rfloor})$ vertices. There are $O(n^{d-1})$ choices for the $(d-1)$-tuple of obstacles, yielding at most $O(n^{d-1}m^{\lfloor d/2 \rfloor})$ vertices in total. Hence there exists a path with $O(n^{d-1}m^{\lfloor d/2 \rfloor})$ turns as claimed. □

For $d = 2$ it is known that that the combinatorial complexity of the union of n polygons in $\mathbb{R}^2$ with a total of M edges is $O(n^2 + Mlogn)$, and similarly in $\mathbb{R}^3$, the union of n polytopes with a total of M facets has complexity $O(n^3 + Mn\log n)$. These bounds for the total complexity of the environment are of approximately the same order as our bound for the path complexity. Both results are due to Aronov and Sharir [AS93], the three-dimensional one being quite recent. It would be interesting to show a similar result also in higher dimensions.

It would be tempting to conjecture that the relatively crude $O(n^{d-1})$ upper bound on the number of facets of the expanded polytopes can be improved, using some clever construction for these polytopes (rather than choosing the separating halfspaces arbitrarily as we did in Lemma 4). However, we have an example in $\mathbb{R}^3$ which rules out this direct way of improvement: One can construct $O(n)$ convex polytopes in $\mathbb{R}^3$, such that none of them can be enlarged without creating new triple intersections, and the combinatorial complexity of the union is n^4 (even if we only count the vertices defined by an edge of one polytope and a facet of another). We omit the description of this example.

As for the actual path complexity, we have a construction showing that the bound in Theorem 5 cannot be improved below $\Omega(n^d)$; in particular, the result is tight in the plane. The construction proceeds by induction on the dimension d. Consider first the planar version (see Figure 1 for $n = 4$). We define n^2 points A_{ij}, $i, j = 0, 1, \ldots, n-1$, on n rays starting in the origin: $A_{ij} = (1 + \varepsilon i)(\cos \frac{2\pi j}{n}, \sin \frac{2\pi j}{n})$, for $i, j = 0, \ldots, n-1$ and $\varepsilon > 0$ sufficiently small. Let obstacle B_i be the convex hull of $A_{i0}, A_{n-i,1}, A_{i,2}, A_{n-i,3}, \ldots$. The union of these n obstacles looks like a star. We can now fill the n concave niches by other obstacles, leaving a small pathway around the inner star. The length of this pathway is n^2, which proves an $\Omega(n^2)$ lower bound for the planar intersecting case.

Now suppose that we have a set of n obstacles, a starting position p and a certain goal position q in $\mathbb{R}^d$, such that any path between a p and q has $\Omega(n^d)$ links. We extend each obstacle to $\mathbb{R}^{d+1}$ by taking the Cartesian product of the obstacle with, say, the interval $[0, 1]$. Let us call the extra coordinate the x_{d+1}-coordinate. We now add n new obstacles, which are the hyperplanes $h_i : x_{d+1} = i/n$, $i = 1, \ldots, n$. The hyperplanes h_i partition $\mathbb{R}^{d+1}$ into $n+1$ slabs. Observe that if we restrict our attention to one of the hyperplanes h_i we have exactly the d-dimensional situation. Let p_i and q_i be the points on h_i that correspond to the starting and goal position for the d-dimensional situation. If i is odd we punch a small hole in h_i around p_i, and if i is even we punch a small in h_i around q_i. Now it is easy to see that any path from p_1 to q_n needs $\Omega(n^{d+1})$ links: it must first go from p_1 to q_2 using $\Omega(n^d)$ links, then it must return to p_3 using another $\Omega(n^d)$ links, and so on. Notice that the punctured hyperplanes are not convex, but they can be replaced by three (intersecting) half-hyperplanes. It is straightforward to adapt the construction so that it

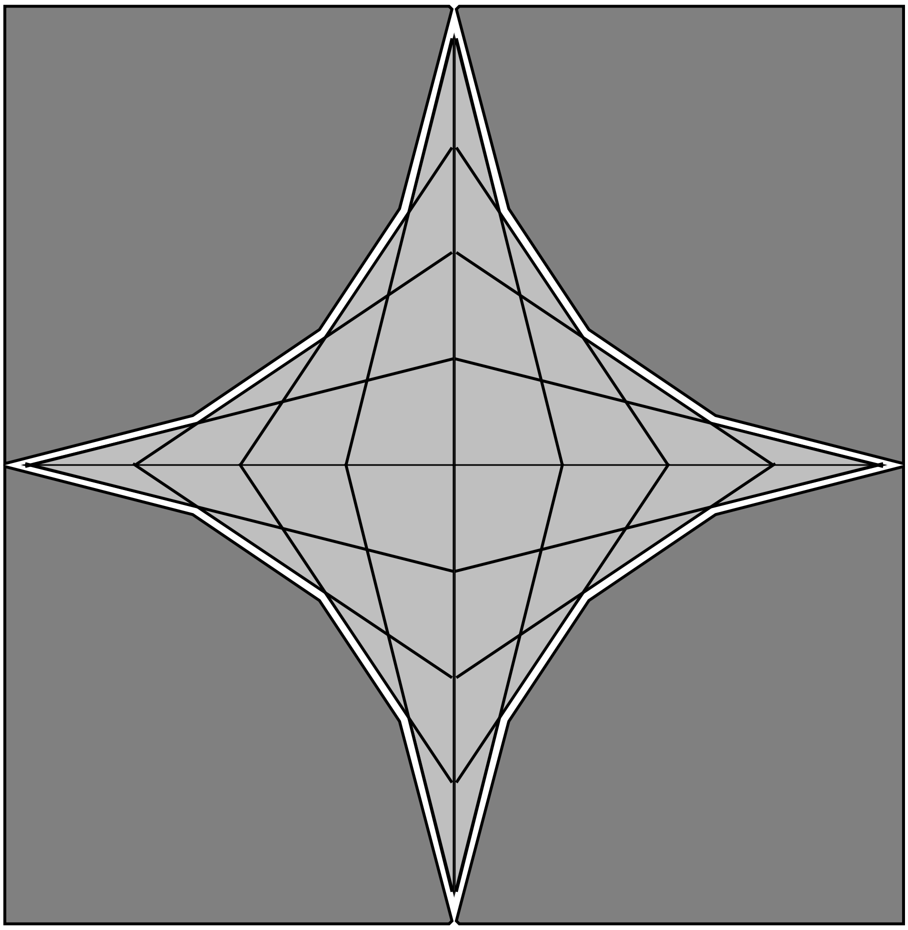

Figure 1: A lower bound example

only uses bounded obstacles. The total number of obstacles used remains still $O(n)$ for any fixed d.

5 Bounds and algorithms for the planar case

We now consider the planar case of the problem, and we turn our attention to the algorithmic aspect.

Touching planar obstacles. Here we consider a family $\mathcal{B}$ of n closed convex obstacles with disjoint interiors. We will show that the path complexity is linear in this case, and we give an efficient algorithm.

Our algorithm proceeds as follows: For every obstacle $B \in \mathcal{B}$, we identify a leftmost and a rightmost point of B, and we let s_B be the segment connecting those two points. If B is unbounded, the line segment may degenerate to a semi-infinite ray, or even to a line. We then compute the trapezoidal map defined by this set of line segments[3]. This

[3]A trapezoidal map induced by a set of line segments is the partition of the plane obtained as follows: through every endpoint, we add a vertical segment not intersecting the other segments and maximal with

trapezoidal map consists of at most $3n+1$ trapezoids. Every (bounded) trapezoid has two vertical sides, and it is bounded from above and below by line segments corresponding to two of the obstacles. It is easy to verify that the interior of a trapezoid is intersected by these two obstacles only. Since they are convex we can separate them by a line. Any point inside the trapezoid and outside the obstacles can be connected to this line by a vertical line segment. This means that any two points inside the trapezoid can be connected by at most three links, provided that they can be connected at all inside the trapezoid. (This need not be the case, because the two obstacles inside the trapezoid are allowed to touch.)

Consider a path γ connecting start and goal positions p and q. We can assume that γ passes through every trapezoid at most twice, since every trapezoid intersects at most two connected components of $\mathbb{R}^2 \setminus (\mathrm{int}\, \overline{\bigcup \mathcal{B}})$. Within a trapezoid, we can replace the portion of γ by at most three links. Hence, there is a path using a linear number of links.

We now turn this existence proof into an efficient algorithm. Consider, for every trapezoid, the two points where the line separating its two obstacles intersects its vertical sides (they are necessarily outside the obstacles). We connect p, q to such points in their respective trapezoids, and we find the path using a graph search algorithm.

In this algorithm, we suppose that the obstacles are described by an oracle. It is sufficient that each of the following operations can be performed by the oracle (in a worst-case time Q): finding the segment s_B for a given B, finding a line separating given obstacles B, B', and finding the common part of their boundaries when they touch. We thus obtain

Theorem 7 *Let $\mathcal{B}$ be a set of n convex obstacles in the plane described by an oracle as above, let p,q be two points. Then, in $O(n(Q + \log n))$ time, we can construct a p-q path with $O(n)$ links avoiding $\mathcal{B}$ or determine that no p-q path exists.*

In particular, if the obstacles are polygons with m vertices in total, one can implement the oracle operations in time $Q = O(\log m)$, assuming that the vertices of each polygon are stored in an array in a clockwise sorted order[DK85]. Hence, a path with $O(n)$ links can be found in $O(n \log m)$ time in this situation.

The motion planning problem for a convex robot. Here we come back to the problem which motivated our research. Given an open convex robot R and a collection $\mathcal{B}^* = \{B_1^*, \ldots, B_n^*\}$ of disjoint open convex obstacles in the plane, as well as positions p and q, find a path that translationally moves R from p to q. It is well-known that this problem can be reduced to the case of a point robot by replacing $\mathcal{B}^*$ by $\mathcal{B} = \{B_1, \ldots, B_n\}$, where $B_i = B_i^* - R = \{b - x \mid b \in B_i^*, x \in R\}$ is the Minkowski difference of B_i^* and the robot. We thus obtain a new set of obstacles in the so-called configuration space. Any point x in this space corresponds to a certain position of the robot; we denote the robot at this position by $R(x)$. The complement of the union of the new obstacles, i.e. the region $\mathbb{R}^2 \setminus (\mathrm{int}\, \overline{\bigcup \mathcal{B}})$, is called the *free space*; we denote it by FP.

Kedem et al. [KLPS86] have shown that the obstacles in $\mathcal{B}$ are *pseudo-discs*, i.e. their boundaries intersect in at most two points and the total number of such reflex vertices on the boundary of the union of $\mathcal{B}$ is $O(n)$. This allows us to compute a linear-size "trapezoidal" map for FP. This map is defined as follows. Consider the points on ∂FP which are either the leftmost or the rightmost point of some B_i, or a reflex vertex of ∂FP which

respect to this property.

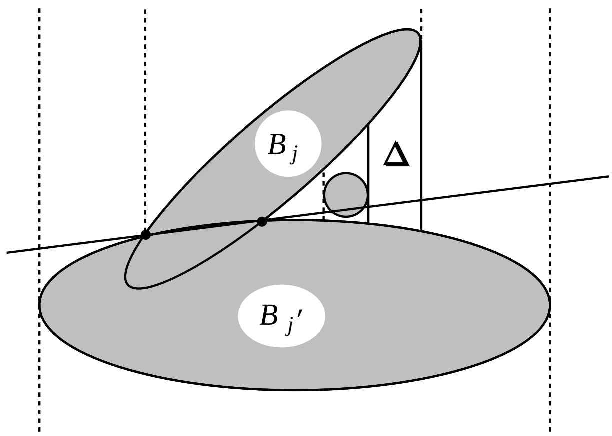

Figure 2: Example of a trapezoidal map.

is not locally extreme in x-direction. (The latter type of points are intersections between boundaries of obstacles.) For each such point we add a vertical segment of maximal length that contains the point and is contained in FP. Observe that each trapezoid Δ in the map is bounded by one or two obstacles (one bounding it from above, and one from below) and by one or two vertical segments. See Figure 2 for an illustration. (The line through the intersection points of ∂B_j and $\partial B_{j'}$ in this figure will be used later.) ¿From the trapezoidal map we then compute a path using a graph search algorithm, in about the same way as in the touching case.

Next we describe the algorithm in more detail. As before, we assume that the obstacles are given by suitable oracles. More precisely, we need two oracles, one for the original bodies B_i^* and the original robot R, and one for the Minkowski differences B_i. The first oracle should be able to answer the following questions:

- given i, find the segment $s_{B_i^*}$ (as defined in the previous subsection),
- given i and a certain placement of R, decide whether B_i^* intersects R,
- given a certain placement of R and a vertical line l, compute $R \cap l$,
- given i and a vertical line l, compute $B_i^* \cap l$.

For the Minkowski differences B_i we need:

- given i and a vertical line l, compute $B_i \cap l$,
- given i, j, decide whether B_i, B_j are disjoint and if yes, return a separating line, if not, return the intersections of their boundaries.

Let Q^* be the worst-case time the oracle on the original bodies and the robot needs, and let Q be the time for the oracle on the Minkowski differences.

The first phase of the algorithm—the computation of the trapezoidal map—is done using a randomized incremental algorithm that follows the abstract framework of Boissonnat et al. [BDS+92]. The *regions* in their framework are the trapezoids; they are defined by at most four obstacles. For the reader's convenience, we briefly sketch the way the algorithm works in our case. (Let us remark that a randomized incremental algorithm to compute the union of a set of pseudo-discs has already been presented by Miller and Sharir [MS90]. Their algorithm is basically the same as the one that we sketch next. They assume, however, that the pseudo-discs have constant complexity, so we cannot use their result directly.) The algorithm adds the obstacles B_i in random order, and it maintains the trapezoidal map of the already added obstacles. When the last object has been added the trapezoidal map that we seek is complete. In a generic step in this algorithm we have to find the trapezoids in the current map that are intersected by (in the terminology of [BDS+92]: are *in conflict with*) the newly added obstacle, and we have to create the new trapezoids that arise. (Note that in the creation phase some trapezoids may have to be merged; to this end one has to maintain the adjacency relation between the trapezoids.) The basic operation in the creation phase is to compute the intersection points of the boundary of a trapezoid in the current map with the new obstacle. A constant number of calls to the oracle for the Minkowski differences is sufficient to compute these intersection points. Locating the intersected trapezoids is done by searching the history. Here the basic operation is to test whether the new obstacle intersects a given trapezoid in the history. We could, of course, simply use the oracle for the B_i and compute the intersection points of the new obstacle with the trapezoid. But we can also test this using calls to the—possibly cheaper—oracle for the original bodies only.

Lemma 8 *It is possible to test in $O(Q^*)$ time whether an obstacle B_i intersects a trapezoid Δ.*

Proof: By definition, B_i intersects Δ if and only if there is a point $x \in \Delta$ such that $R(x)$ intersects B_i^*. So testing whether B_i intersects Δ amounts to testing whether B_i^* intersects the sweeping area $\text{Sweep}(\Delta) = \bigcup\{R(x) : x \in \Delta\}$.

Let B_j and $B_{j'}$ be the two obstacles that bound Δ from above and from below, respectively. Let x_l and x_l' be the upper and lower endpoint of the left boundary segment of Δ. Define x_r and x_r' analogously for the right boundary segment of Δ. Notice that it is possible that $x_l = x_l'$ or $x_r = x_r'$ (but not both), but this does not change the following argument. A simple example for a disc-shaped robot is given in Figure 3. In this figure, the boundary of Δ is drawn fat and dashed, whereas the boundary of $\text{Sweep}(\Delta)$ is fat and solid.

One easily verifies that $\partial\text{Sweep}(\Delta)$ consists of parts of the the boundaries of B_j^*, $B_{j'}^*$, $R(x_l)$, $R(x_l')$, $R(x_r)$ and $R(x_r')$, together with the vertical line segment s_l that connects the leftmost point of $R(x_l)$ to the leftmost point of $R(x_l')$, and the vertical line segment s_r that connects the rightmost point of $R(x_r)$ to the rightmost point of $R(x_r')$. We first test whether B_i^* intersects $R(x_l)$, $R(x_l')$, $R(x_r)$ or $R(x_r')$, using our oracle. If an intersection is found we are done. Otherwise we proceed as follows. Cut $\partial\text{Sweep}(\Delta)$ into an upper part β and a lower part β' by removing the segments s_l and s_r. Because the original obstacles are disjoint and we already tested B_i^* for intersections with $R(x_l)$, $R(x_l')$, $R(x_r)$ and $R(x_r')$, we know that B_i^* does not intersect β or β'. Note that $\text{Sweep}(\Delta)$ is monotone, that is, any vertical line intersects it in a single—possibly empty—interval. Hence, B_i^* intersects $\text{Sweep}(\Delta)$ if and only if it hits β when it is translated upward, but it does not hit β'. Next we describe how to test this.

Without loss of generality, let us consider the upper chain β. To test whether B_i^* will hit $\beta \cap \partial B_j^*$ we first try to find a vertical line that intersects both B_i^* and $\beta \cap \partial B_j^*$:

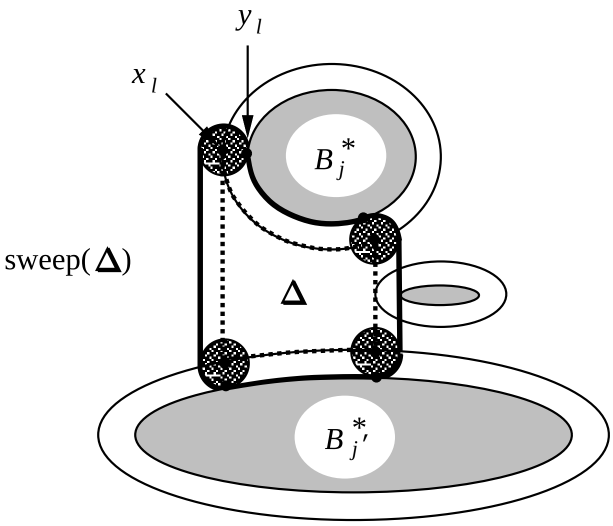

Figure 3: The sweeping area of a trapezoid.

this can be done in constant time after computing the segment $s_{B_i^*}$, since we know the extreme points of $\beta \cap \partial B_j^*$. If there is no such line then B_i^* will not hit $\beta \cap \partial B_j^*$. If there is such a line, it remains to compute its intersection points with B_i^* and B_j^* in time Q^*, to check whether B_i^* lies above $\beta \cap \partial B_j^*$ or below. Testing whether B_i^* will hit $\beta \cap \partial R(x_l)$ or $\beta \cap \partial R(x_l')$ is done in a similar way. □

Next we analyze the running time of this algorithm. Recall that a generic step in the algorithm consists of two phases: finding the trapezoids that are in conflict with the newly added obstacle, and creating the new trapezoids. The first phase requires a number of tests whether the obstacle intersects certain trapezoids, each test taking $O(Q^*)$ time. The second phase requires the computation of the actual intersection of some trapezoids with the obstacle, which takes $O(Q^*)$ time. We know that the number of trapezoids in the current map at any point during the algorithm is linear in the number of already added obstacles. It then follows from the general results of Boissonnat et al. [BDS+92] that expectation of the total number of trapezoid-obstacle tests is $O(n \log n)$. It also follows that the expected total number of trapezoids that is generated during the process is $O(n)$. This is easily seen to imply that the expected number of times we have to compute intersections of trapezoids and obstacles is $O(n)$ as well. We conclude that the trapezoidal map can be constructed in $O(n(Q^* \log n + Q))$ randomized time.

After we have computed the trapezoidal map, we proceed as in the touching case: we compute for every trapezoid in the decomposition a separating line, we connect the start and goal positions to the lines in their respective trapezoids by a vertical segment, and we find a path using a graph search algorithm. The only non-trivial operation in this procedure is the computation of a separating line. Consider a trapezoid Δ, and let B_j, $B_{j'}$ be the obstacles that bound Δ from above and below, respectively. If B_j and $B_{j'}$ are disjoint then life is easy: the oracle will provide us with a separating line. But it may very well happen that B_j and $B_{j'}$ intersects, as in Figure 2. Fortunately, the oracle can also be used in this case: the line through the two intersections points of ∂B_j and $\partial B_{j'}$ must separate B_j from $B_{j'}$ inside Δ (this follows from straightforward geometric arguments). See Figure 2.

Hence, the second phase of the algorithm takes $O(nQ)$ time, and we obtain the following result.

Theorem 9 *Let $\mathcal{B}^*$ be a set of n disjoint open convex obstacles, R an open convex robot, p,q two positions of R. Assume that oracles as described above for the original bodies B_i^* and R, and for the Minkowski differences B_i are available. Then in randomized time $O(n(Q^* \log n + Q))$, one can either compute a translational motion of R between p and q consisting of $O(n)$ links, or determine that no p-q translational motion is possible.*

Let us consider the case where the obstacles are convex polygons with m vertices in total and R is a convex k-gon. We assume that we are given arrays that store the vertices of the polygons in order. Dobkin and Kirkpatrick [DK85] have shown that in that case the oracle for the original bodies B_i^* and R can be implemented to work in $O(\log(m + k))$ time. We now turn our attention to the oracle on the Minkowski differences.

Lemma 10 *The oracle for the Minkowski differences can be implemented such that it works in $O(\log^2(k + m))$ time.*

Proof: The first operation that the oracle must be able to perform (given i and a vertical line l, compute $B_i \cap l$) can be performed in the same way as the second operation—in fact, it is simpler—so we concentrate on the latter.

Let us first consider the case where B_i and B_j do not intersect, and we want to find a separating line. As noted above, Dobkin and Kirkpatrick [DK85] have given an algorithm for computing a separating line for two convex polygons in logarithmic time. However, we cannot afford to compute each B_i explicitly. A closer look at their algorithm reveals that it roughly works as follows. A chain of edges on each polygon is maintained that contains the portion of the boundary that is still relevant. At each step of the algorithm the median edges of these chains are considered. Depending on the positions of these edges and their vertices, half of one of the chains can be eliminated from further consideration. Notice that we need not use the exact median edge of a chain; it is enough if we have an edge that splits the chain into two parts of almost equal size. Hence, to use the algorithm in our case it suffices to be able to produce such an edge for a chain on B_i. This can be done in logarithmic time, as follows. Consider a chain β on the boundary of B_i; the ordered sequence of edges of β corresponds to a merge of the ordered sequence of edges of certain chains β' and β'' on B_i^* and R, respectively. Take the median edge e on the longer of the two chains, say β', and search with (the slope of) this edge in the other chain β''. In this way we obtain the edges in β that are adjacent to the edge of β corresponding to e and, hence, the endpoints of this edge. (Note that the adjacent edges can be edges of β' and β''.) The number of edges on either side of e in β is no more than $|\beta'|/2 + |\beta''| \leqslant 3|\beta|/4$. We conclude that the time needed for a basic operation in the algorithm of [DK85] is $O(\log(k + m))$, leading to a total time for the oracle of $O(\log^2(k + m))$. In the forgoing we have assumed that B_i and B_j are disjoint. If we run the algorithm of Dobkin and Kirkpatrick on two intersecting polygons, it will return a point x in $B_i \cap B_j$. However, our oracle should provide some more information, namely the intersection of the boundaries of B_i and B_j. (Note that because B_i and B_j are Minkowski differences, $\partial B_i \cap \partial B_j$ consists of two points or one line segment. To simplify the discussion, we will assume in the remainder that the intersection consists of exactly two points; the adaptation to the general case is completely straightforward.) First, we compute the intersections of the horizontal line through x with ∂B_i and ∂B_j. This can again be done in $O(\log^2(m+k))$ time by binary search. Note that two of the four points that we find must be points that lie on the boundary of one of the polygons, and are contained in the other. Let q be such a point and assume without loss of generality that it lies on ∂B_i and inside B_j. Next, we find a point $q' \in \partial B_i$ that lies outside B_j. Observe that point q' corresponds to a placement of R where it touches B_i^* and misses B_j^*. Hence, we can find q' by computing in logarithmic time a line that separates B_i^* and B_j^*, and placing R such that it touches B_i and does not intersect this line. (Such a placement is easy to find in logarithmic time.)

So we have two points available on ∂B_i, one inside B_j and one outside B_j. Since we know that ∂B_i and ∂B_j intersect twice we can now use binary search to find the intersection points. The basic operation during the binary search is this: given a point on ∂B_i, is the point inside B_j or not? This question amounts to checking whether a certain placement of R intersects B_j^* and, hence, it can be answered in $O(\log(m+k))$ time. It follows that the binary search for the intersection points of ∂B_i and ∂B_j takes $O(\log^2(m+k))$ time in total. ◻

We remark that the lemma also holds if the obstacles and the robot are convex figures bounded by Jordan arcs, instead of straight line segments; this immediately follows from the fact that Dobkin and Kirpatrick's technique also works (with a few trivial adaptations) in this situation. Thus the translational motion planning can be performed in $O(n \log^2(k + m))$ time, yielding a path with $O(n)$ links. Note that the complexity of the union of the polygons in $\mathcal{B}$ can be as high as $\Theta(kn+m)$.

General obstacles. For the general case of arbitrary convex obstacles, we proceed similarly as in Section 4. We replace the family $\mathcal{B}$ by another family $\mathcal{C}$. For every pair $B_i, B_j \in B$ with B_i and B_j disjoint, we find a separating line ℓ_{ij}. We also compute lines ℓ_i^p and ℓ_i^q separating B_i from p and q. We then replace B_i by C_i defined as the intersection of the halfspaces bounded by the ℓ_{ij}, ℓ_i^p and ℓ_i^q. By Lemma 2, p and q can be connected among the B_i exactly if they can be connected among the C_i. Every C_i is an open convex polygon with at most $n+1$ edges. A shortest path between p and q only bends at convex vertices of the union of the C_i. Each such vertex is a vertex of some polygon in $\mathcal{C}$. Hence, there are at most $O(n^2)$ such bends in the shortest path.

In order to compute such a path, we consider the family of all edges of the polygons of $\mathcal{C}$, and consider the arrangement of these $O(n^2)$ line segments. The points p and q can be connected iff they lie in the same cell of this arrangement. This cell has complexity $O(n^2\alpha(n))$ and can be computed in expected time $O(n^2\alpha(n)\log n)$, where $\alpha(n)$ is the functional inverse of the Ackermann-function, see [CEG+91]. Once the cell is computed, it is not difficult to compute a path with $O(n^2)$ links in time linear in the complexity of the cell.

For the oracle describing the obstacles, this time we require that it can return a separating line for two obstacles (or say that they intersect) and also separate p, q from the obstacles. If Q denotes the running time for an oracle call, we get

Theorem 11 *Let $\mathcal{B}$ be a set of n convex obstacles in the plane described by an oracle as above, let p and q be two points. In (expected) time $O(n^2Q + n^2\alpha(n)\log n)$, one can compute a p-q path with $O(n^2)$ links or determine that no p-q path exists.*

For polygons with m vertices in total, the oracle is implemented in $O(\log m)$ time, yielding total expected time $O(n^2\alpha(n)\log n + n^2\log m)$.

6 Disjoint obstacles in higher dimensions

Using the results on the planar case from the previous section, it is not difficult to prove the following:

Theorem 12 *Let $\mathcal{B} = \{B_1, \ldots, B_n\}$ be a collection of pairwise disjoint possibly unbounded open convex bodies. Then any two points in the same connected component of $\mathbb{R}^d \setminus \bigcup\mathcal{B}$ can be connected by a path consisting of $O(n)$ segments.*

Proof: If all obstacles are bounded, the proof is very easy. Indeed, in this case none of the obstacles can separate $\mathbb{R}^d$, and the complement has only one component. We can fix any plane π passing through the given points p, q, and connect these points by a path lying within π. Hence we may use the planar result from Section 5.

The situation becomes slightly more complicated if we also allow unbounded obstacles, since then the complement may be disconnected. It can also happen that the two points p and q can connected in $\mathbb{R}^d$, but not within an arbitrarily chosen hyperplane containing p and q.

We will use two easy facts, whose formal proof we omit. First, if $B_1, \ldots, B_n$ are open, convex and disjoint and separate p from q, then there is a single B_i also separating p from q. Second, an open convex obstacle separating p from q must be an open slab, bounded by two parallel hyperplanes.

Thus, if we have a hyperplane h through p, q such that p, q cannot be connected within h, there must be one obstacle B_i "responsible" for it. We show that one obstacle can only be responsible for one hyperplane. Suppose not, let B_i separate p from q in two hyperplanes h, h'. Then both $B_i \cap h$, $B_i \cap h'$ are slabs (in h, h', respectively), and it is easy to see that the convex hull of such two $(d-1)$-dimensional slabs contains a hyperplane separating .

Thus, there are at most n bad hyperplanes and a path exists within any hyperplane through p, q different from these. Then, by induction on the dimension we may assume that such a hyperplane contains a path of $O(n)$ links, and we are done. $\square$

It is easy to construct n unbounded convex sets and two points p, q, such that any path between p and q needs a linear number of links, thus Theorem 12 is optimal. For completeness, we also give an example of n disjoint *bounded* convex obstacles that enforce a linear number of links. Consider a set of $n/2d$ hypercubes $C_1, \ldots, C_{n/2d}$, each centered around the origin, such that $C_1 \subset C_2 \subset \cdots \subset C_{n/2d}$. We make C_{i+1} relatively large with respect to C_i. The set of obstacles in our example is the set of facets of the hypercubes. However, the facets of a hypercube touch each other, so we need to modify them slightly. This is done as follows. Call the facets of a hypercube which are orthogonal to the x_d-axis its x_d-facets. The x_i-facets of the cubes are shrunk in the x_{1+1}- up to x_d-direction by a

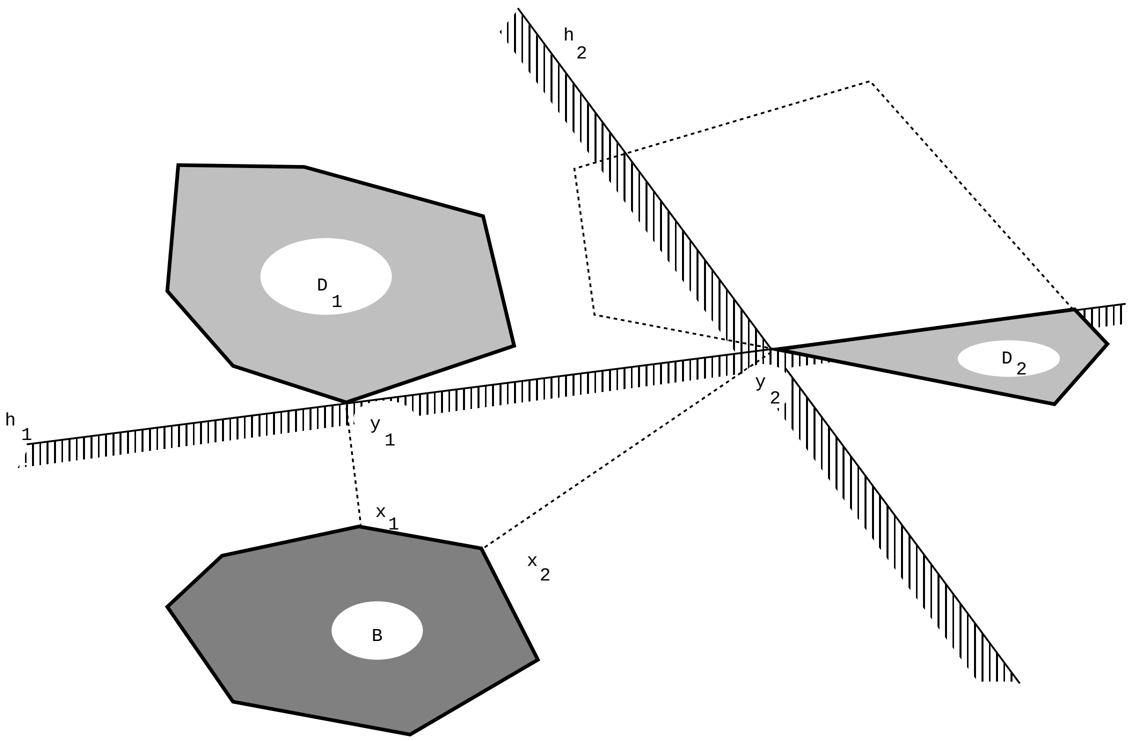

Figure 4: Lower bound for the disjoint case.

small amount ε, and they are extended by 2ε in the x_1- up to x_{i-1}-direction. If we do this for each of the hypercubes C_i we end up with a set of n disjoint convex obstacles. See Figure 4 for a planar example of the construction. It is easy to see that it takes $\Omega(n)$ links to go from the origin to a point outside the largest cube $C_{n/2d}$.

7 Conclusion and Extensions

We regard the current work as an initial study of the considered problems, and many questions remain open. It would be nice to tighten the combinatorial bounds for dimensions 3 and more; most notably, there is a large gap between the bounds for the translational motion of a convex robot among convex obstacles in $\mathbb{R}^3$, and a new idea seems to be needed for improving the upper bound.

There are also some other intermediate cases to be considered. For instance, the motion planning problem for a point robot moving among disjoint convex obstacles is not very realistic, since by our definitions the robot can move even through infinitely thin passages. It would be more natural to allow for disjoint but touching obstacles, where the infinitely thin passages are sealed up. We have some preliminary results for this case in higher dimensions (the planar case being subsumed by the results of the present paper).

The algorithmic aspects in $\mathbb{R}^3$ are almost untouched. Also it remains to see to what extent our approach can help in handling practical motion planning problems.

One might ask about the possibility of a further generalization of our results to, say, translational and rotational movement of a convex robot among convex obstacles. It turns out that no upper bound on the number of "simple" pieces of the movement can be given

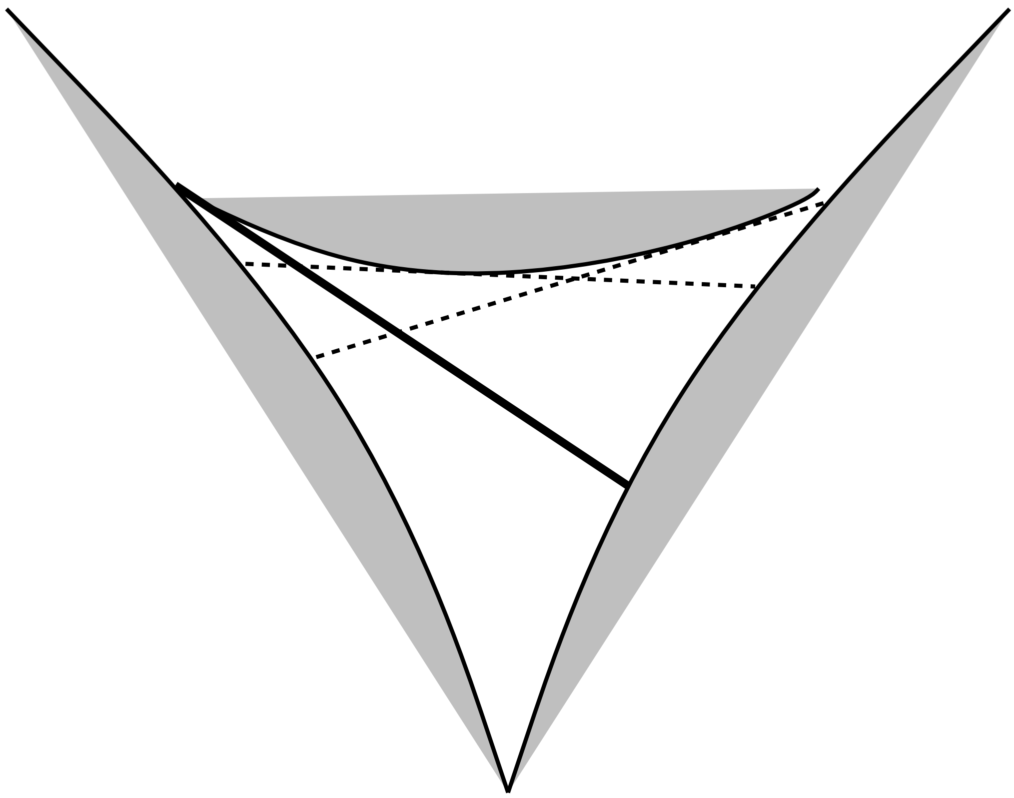

Figure 5: A rod moving among convex obstacles.

for arbitrary convex obstacles (where "simple" would mean defined by bounded degree algebraic curves). This can be seen by considering a rod (a segment in the plane) among obstacles arranged in such a way that the rod has to move along a unique curve (in the free space) enforced by a triple contact with three suitably shaped convex obstacles. See Figure 5. Such a curve can only be described in terms of the obstacles, and thus such problems need a somewhat different approach.

Acknowledgment We would like to thank E. Vogt for his great help in the technical realization of the proof of Lemma 2.

References

[AY89] H. Alt and C. K. Yap. Motion planning in *cl*-environments: A case for a realistic model. In *Proc. 1st Workshop Algorithms Data Struct. (WADS)*, volume 382 of *Lecture Notes in Computer Science*, pages 373–380. Springer-Verlag, 1989.

[AS93] B. Aronov and M. Sharir. Preliminary paper draft and a lecture of B. Aronov at Dagstuhl Seminar – Computational Geometry, March 1993.

[BDS+92] J.-D. Boissonnat, O. Devillers, R. Schott, M. Teillaud and M. Yvinec. Applications of random sampling to on-line algorithms in computational geometry. *Discrete Comput. Geometry* 8:51–71, 1992.

[CD85] L. P. Chew and R. L. Drysdale, III. Voronoi diagrams based on convex distance functions. In *Proc. 1st Annu. ACM Sympos. Comput. Geom.*, pages 235–244, 1985.

[CD88] J. Canny and B. R. Donald. Simplified Voronoi diagrams. *Discrete Comput. Geom.*, 3:219–236, 1988.

[CEG+91] B. Chazelle, H. Edelsbrunner, L. Guibas, M. Sharir, and J. Snoeyink. Computing a face in an arrangement of line segments. In *Proc. 2nd ACM-SIAM Sympos. Discrete Algorithms*, pages 441–448, 1991.

[DK85] D. P. Dobkin and D. G. Kirkpatrick. A linear algorithm for determining the separation of convex polyhedra. *J. Algorithms*, 6:381–392, 1985.

[KLPS86] K. Kedem, R. Livne, J. Pach, and M. Sharir. On the union of Jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete Comput. Geom.*, 1:59–71, 1986.

[KM91] T. C. Kao and D. M. Mount. An algorithm for computing compacted voronoi diagrams defined by convex distance functions. In *Third Canadian Conference on Computational Geometry*, pages 104–109, 1991.

[Lat91] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991.

[MS90] N. Miller and M. Sharir, Efficient randomized algorithm for constructing the union of fat triangles and of pseudodiscs. Unpublished manuscript.

[Mun84] J. R. Munkers. *Elements of algebraic topology*. Addison-Wesley, 1984.

[Rot88] J.J. Rothman. *An introduction to algebraic topology (GTM 119)*. Springer-Verlag, 1988.

[Sif89] S. Sifrony. A real nearly linear algorithm for translating a convex polygon. Technical Report 476, New York University, 1989.

[SS90] J. T. Schwartz and M. Sharir. Algorithmic motion planning in robotics. In J. van Leeuwen, editor, *Algorithms and Complexity*, volume A of *Handbook of Theoretical Computer Science*, pages 391–430. Elsevier, Amsterdam, 1990.