Efficient collision detection for moving polyhedra

Elmar Schömer* Christian Thiel†

Abstract

In this paper we consider the following problem: given two general polyhedra of complexity n, one of which is moving translationally or rotating about a fixed axis, determine the first collision (if any) between them. We present an algorithm with running time $O(n^{8/5+\epsilon})$ for the case of translational movements and running time $O(n^{5/3+\epsilon})$ for rotational movements, where ϵ is an arbitrary positive constant. This is the first known algorithm with sub-quadratic running time.

1 Introduction

The demands on quality, security and higher production capacity in manufacturing increase the need for planning during the phase of product design. To find potential faults in the design as soon as possible one uses simulation programs: these predict the physical properties and reactions of the product and check whether particular prefabricated parts can be easily assembled. For the latter purpose, the simulation of assemblies and robots, efficient methods for *collision detection* are needed. In general collision detection is an essential prerequisite of simulations of mechanical tools.

Regarding the significance of this problem we consider in our paper efficient algorithms for collision detection. It is known ([Bo79, Ca84]) that in $\mathbb{R}^3$ a collision between a moving polyhedral object and a stationary obstacle is computable in time $O(n^2)$. Here, n denotes the complexity of the two objects, i.e. the number of vertices, edges and faces. We attempt to solve this problem in sub-quadratic time. Our results are justified by the following model: Objects are rigid bodies (polyhedra) in $\mathbb{R}^3$, their surfaces consist of planar faces with straight boundaries. An object may be moving translationally in an arbitrary direction or it may be rotating about an arbitrary axis. These restrictions are based on the fact that real objects can be easily modelled by polyhedra and every motion can be approximated by a sequence of translations and rotations. As our model of computation we take the standard Real-RAM-model ([PS88]).

1.1 Previous results

There are (up to now) only efficient solutions for some special cases of **translated** polyhedra. Dobkin and Kirkpatrick demonstrate in [DK85, DK90] the efficiency of the hierarchical representation for solving distance problems between convex polyhedra: using this data structure one can determine the collision between two convex polyhedra in time $O(\log^2 n)$. If one of the objects is not convex, an algorithm with running time $O(n \log n)$ is possible (for more details see [DHKS90, Sch94]). The collision between a translationally moving and a stationary c-iso-oriented polyhedron can be computed in time $O(c^2 n \log^2 n)$ (see [Sch94]). Also in this work the first sub-quadratic collision detection algorithm for two general polyhedra (one of which is translated and one is stationary) is developed.

For the case of **rotations** there are no sub-quadratic algorithms known. Even the special case of two convex polyhedra, one of which is rotating has not been solved up to now. This particular problem was posed as an open question by Jack Snoeyink during the Third Dagstuhl Seminar on Computational geometry in March 1993:

Given two convex polyhedra A,B, and an axis

*Universität des Saarlandes, Fachbereich 14, Informatik, Lehrstuhl Prof. Hotz, Im Stadtwald, D-66041 Saarbrücken, Germany. E-mail: schoemer@cs.uni-sb.de.

†Max-Planck-Institut für Informatik, Im Stadtwald, D-66123 Saarbrücken, Germany. E-mail: thiel@mpi-sb.mpg.de. This author was supported by the ESPRIT Basic Research Actions Program, under contract No. 7141 (project ALCOM II).

of rotation, compute the smallest angle by which B has to rotate to meet A. Can this be done in sub-quadratic time?

1.2 New results and outline

In this paper we give the first sub-quadratic algorithms, which solve the collision problem between two **general** polyhedra, one of which is moving translationally or rotating about a fixed axis, whereas the other is stationary. In particular we get an upper bound of $O(n^{8/5+\epsilon})$ for the translational movement and $O(n^{5/3+\epsilon})$ for the rotational movement*.

The first collision between two polyhedra can either be a collision between a vertex of one polyhedron and a facet of the other or a collision between two edges. The former case is the simpler one and will be treated in the last part of the paper by plane sweep techniques (see Section 6). The latter problem is the harder problem and we concentrate on it. We show how to preprocess the set of stationary segments, such that we can efficiently compute the first segment hit by a moving query segment. We proceed in three steps: In the first step we use the parametric search technique of Meggido (see [Meg83]) to reduce the problem of computing the first intersection during the motion to the problem of computing the total number of intersections during the motion. In the second step we show how to reduce the latter problem to a combination of halfspace and simplex range searching problems; the key technique here is linearization, which was for the first time suggested in [YY85]. In the third step we solve the range searching problems using known techniques of van Kreveld and Matoušek. After that description our general technique will be applied to the collision problem of line segments which move translationally or rotate about a fixed axis: in Section 4 we will deduce the needed appropriate linearizations and get an upper bound of $O(n^{5/3+\epsilon})$ for both kinds of motions. Applying a recent result of Pellegrini [Pel93] in Section 5 we sketch an improved solution for translationally moving edges with running time $O(n^{8/5+\epsilon})$. Section 6 considers the collision problem for facets and vertices.

2 General collision detection and parametric search

Let $\mathcal{T}$ be a class of (topologically closed) geometric objects, i.e. closed subsets of $\mathbb{R}^d$, and let $\mathcal{S}$ be some set of n objects in $\mathcal{T}$. Let $\mathcal{Q}$ be another class of (topologically closed) geometric objects in $\mathbb{R}^d$. Further let $\mathcal{M}$ be a set of *admissible motions* for the objects in $\mathcal{Q}$, i.e. in our case the set of all possible translations respectively rotations.

For an object S of $\mathcal{T}$ and an object $Q \in \mathcal{Q}$, which moves according to a formula ℓ, we denote the first time, such that S is hit by Q, with $\phi(S, Q, \ell)$. If there is no such collision we set $\phi(S, Q, \ell) = \infty$. Our goal is to build a data structure that, given a query object $Q \in \mathcal{Q}$ and the equation $\ell \in \mathcal{M}$ of a motion, computes quickly $\phi(\mathcal{S}, Q, \ell) := \min_{S\in\mathcal{S}} \phi(S, Q, \ell)$ together with a $S \in \mathcal{S}$ such that $\phi(S, Q, \ell) = \phi(\mathcal{S}, Q, \ell)$. We call this the *online collision problem for $\mathcal{Q}$ with respect to $\mathcal{T}$.*

Suppose we have an efficient algorithm A_s that, given a query object $Q \in \mathcal{Q}$ and a motion $\ell \in \mathcal{M}$, decides in T_s time, whether the moving object intersects some objects of $\mathcal{S}$ within a given time period $[0, t]$. In our case this time period is represented by the length of a translation or by the angle of a rotation. We also assume that the algorithm can detect the case when exactly one object of $\mathcal{S}$ is intersected and that it can identify this object. We call such a procedure an *emptiness algorithm.* Using this emptiness algorithm we can easily decide if a given time t is less, equal or greater than $\phi(\mathcal{S}, Q, \ell)$. Meggido's parametric search technique (see [Meg83]) replaces A_s by a parallel algorithm A_p that uses P processors and runs in T_p parallel time. Then it simulates A_p generically on the unknown value $t^* := \phi(\mathcal{S}, Q, \ell)$ and delivers an algorithm that computes t^* in time $O(PT_p + T_sT_p \log P)$.

3 The emptiness algorithm

Our strategy is to reduce the collision problem to a problem for other objects that do not move and then solve the latter by known techniques. We will proceed in two steps. Firstly we linearize the problem and construct a multilevel data structure for counting all collisions (respectively for testing, if there is any collision) within a given time interval. Then we modify this algorithm and get the emptiness algorithm needed for the parametric search technique.

In many applications one (complicated) query problem can be expressed as the combination of several other (easier) query problems. A general notion for the composition of general query problems was introduced in [Krev92]:

Let $\mathcal{P} = \{p_1, p_2, \ldots p_n\}$ be a set of n points in $\mathbb{R}^d$, let $\mathcal{R}$ denote the set of all simplices in $\mathbb{R}^d$, let $\mathcal{S} = \{s_1, \ldots, s_n\}$ be a set of n objects, and let $\mathcal{Q}$ denote a set of queries on $\mathcal{S}$. The *composed query problem* $(\mathcal{S}', \mathcal{Q}')$ is defined as follows: $\mathcal{S}' = \{(p_i, s_i)|\ 1 \le i \le n\}$, $\mathcal{Q}' = \mathcal{R} \times \mathcal{Q}$ and the answer set for a query $(R, Q) \in \mathcal{Q}'$ is given by $\{(p, s)|\ (p, s) \in \mathcal{S}'$ and $p \in R$ and $s \in Q\}$. We also say that $(\mathcal{S}', \mathcal{Q}')$ is obtained from $(\mathcal{S}, \mathcal{Q})$ by *simplex composition.*

*Throughout this paper, ϵ denotes an arbitrary small positive constant.

Simplices in d-space are the intersection of at most $d+1$ many halfspaces. Therefore we can w.l.o.g. consider simplex compositions where the simplices are halfspaces. In this case we also use the term *halfspace composition*.

3.1 General form of linearization

In this section we introduce the concept of linearization. It allows to translate a complicated test in some low dimensional space into a test in some higher dimensional space but involving only linear tests. Here we want to test whether a moving object Q, whose location at time τ is described by $Q(\tau)$ intersects a stationary object S in some time interval $[0,t]$. To find a *linearization* of this problem means to establish the equivalence

$$\begin{aligned} &[\exists \tau \,:\, 0 < \tau < t,\ Q(\tau) \cap S \neq \emptyset] \qquad (1) \\ \iff &\bigvee_{i=1}^{dis} \bigwedge_{j=1}^{con} \left[\sum_{k=1}^{dim} \delta_k^{ij}(Q,t)\, \zeta_k^{ij}(S) \bowtie 0 \right], \end{aligned}$$

where $\bowtie \in \{<,>,=,\leq,\geq\}$, *dis*, *con*, *dim* are positive constants, and $\delta_k^{ij}(Q,t)$ respectively $\zeta_k^{ij}(S)$ are rational functions of constant degree depending on the kind of motion and the kind of objects.

Having such a linearization we map the objects $S \in \mathcal{S}$ into the points $p^{ij} := (\zeta_1^{ij}(S), \zeta_2^{ij}(S), \ldots, \zeta_{dim}^{ij}(S))$ in $\mathbb{R}^{dim}$ and the query object Q into the hyperplanes $h^{ij} := (\delta_1^{ij}(Q,t), \delta_2^{ij}(Q,t), \ldots, \delta_{dim}^{ij}(Q,t))$ in the same space. Then we can think of any $\sum_{k=1}^{dim} \delta_k^{ij}(Q,t)\, \zeta_k^{ij}(S) \bowtie 0$ as the condition, that (depending on $\bowtie$) the point p^{ij} lies on the hyperplane h^{ij} respectively in a halfspace bounded by h^{ij}. Because each conjunction of (1) can be interpreted as the composition of *con* halfspace range searching problems we can find the objects in $\mathcal{S}$ satisfying a particular conjunction by applying halfspace composition *con* times. The disjunctions of (1) correspond to the union of ranges. By rewriting the defining formula, we can assume that these are disjoint unions: a formula $A \vee B$ can be rewritten as $A \vee (B \wedge \neg A)$. Now for counting all objects hit by the moving query object we can just sum up the solutions of the *dis* composed problems (defined by the conjunctions).

In Section 4 we deduce the linearization for the collision problem between a set of moving line segments (all moving in the same direction or all rotating about the same axis) and a set of stationary segments in $\mathbb{R}^3$. There we get constant values *dis*, *con* and *dim*, especially $dim = 5$.

3.2 The data structure

In his Ph.D. thesis [Krev92] Marc van Kreveld investigated efficient solutions for simplex composition† of query problems:

Theorem 1 **([Krev92])** *Let $\mathcal{P}$ be a set of n points in dim-space, and let $\mathcal{S}$ be a set of n objects in correspondence with $\mathcal{P}$. Let T be a data structure on $\mathcal{S}$ having building time $p(n)$, size $s(n)$ and query time $q(n)$. For an arbitrary small constant $\epsilon > 0$, the application of simplex composition on $\mathcal{P}$ to T results in a data structure D of*

1. *size $O(n^{\epsilon}(n^{dim} + s(n)))$ and query time $O(q(n) + \log n)$, or*
2. *size $O(n + s(n))$ and query time $O(n^{\epsilon}(n^{1-1/dim} + q(n)))$, or*
3. *building time $O(m^{\epsilon}(m + p(n)))$, size $O(m^{\epsilon}(m + s(n))$ and query time $O(n^{\epsilon}(q(n) + n/m^{1/dim}))$ for every fixed m such that $n \leq m \leq n^{dim}$,*

assuming that $s(n)/n$ is non-decreasing and $q(n)/n$ is non-increasing. Reporting takes additional $O(k)$ time if there are k answers.

Lemma 2 *If there is a parallel query algorithm for T running in time $t(n)$ using $P(n)$ processors then the query algorithm for the last data structure can be parallelized such that it runs in $O(t(n) + \log n)$ parallel steps using $O(n^{\epsilon}(P(n) + n/m^{1/dim}))$ processors, assuming that $P(n)/n$ and $t(n)/n$ are non-increasing.*

In our case we apply *con* halfspace compositions starting with a halfspace range searching problem. Therefore Theorem 1 leads to a data structure with building time and size $O(m^{1+\epsilon})$, which can count all objects in S satisfying a particular conjunction of (1) in query time $T_s := O(\frac{n^{1+\epsilon}}{m^{1/dim}})$, for every fixed m such that $n \leq m \leq n^{dim}$. We can parallelize that query algorithm such that it runs in $T_p := O(\text{polylog } n)$ parallel time with $P := O(\frac{n^{1+\epsilon}}{m^{1/dim}})$ processors. Using the parametric search technique this gives us the first time t^* of any collision in time $O(PT_p + T_s T_p \log P) = O(\frac{n^{1+\epsilon}}{m^{1/dim}})$. To get the first hit object we start the corresponding reporting algorithm satisfying the same resource bounds.

Theorem 3 *The on-line collision problem with linearization (1) can be solved with a data structure of size and preprocessing time $O(m^{1+\epsilon})$ and query time $O(\frac{n^{1+\epsilon}}{m^{1/dim}})$, for every fixed m such that $n \leq m \leq n^{dim}$.*

Assume we have n moving objects $Q \in \mathcal{Q}$ instead of only one, and we want to determine the first collision

†Actually we use only halfspace composition

between any pair Q, S,for $Q \in \mathcal{Q}$ and $S \in \mathcal{S}$. We apply the solution to the on-line problem and query the data structure of Theorem 3 with each moving element. This gives us a list of n candidates in which we can find the first collision in time $O(n)$.

Using this approach we need $O(m^{1+\epsilon})$ preprocessing time and $n \times O(\frac{n^{1+\epsilon}}{m^{1/dim}})$ query time. To find the best time bound we have to minimize the function

$$c_1 m^{1+\epsilon} + c_2 \frac{n^{2+\epsilon}}{m^{1/dim}},$$

where c_1, c_2 are the O-constants of the resource bounds. This function achieves its minimum for m satisfying $c_1 m^{1+\epsilon} = c_2 \frac{n^{2+\epsilon}}{m^{1/dim}}$ i. e.

$$m = \left(\frac{c_2}{c_1}\right)^{\frac{dim}{dim\epsilon+dim+1}} n^{\frac{(2+\epsilon)dim}{dim\epsilon+dim+1}} = O(n^{\frac{2dim}{dim+1}+\delta}).$$

This proves the following result.

Corollary 4 *Given a subset S of n objects from $\mathcal{S}$ and a set Q of n moving objects from $\mathcal{Q}$. Assume that there is a linearization of the collision problem for $\mathcal{Q}$ with respect to $\mathcal{T}$ in the form of (1). Then we can find in $O(n^{\frac{2dim}{dim+1}+\epsilon})$ time the first collision between any elements of Q and S.*

Corollary 5 *Given two polyhedra of complexity n, one of which is moving translationally respectively is rotating about a fixed axis. The first collision between any two edges of them can be computed in time $O(n^{5/3+\epsilon})$.*

4 Collision of translationally or rotationally moving line segments

Formulation of the problem:
Given: Two line segments l_{ab} and l_{cd} with endpoints **a**, **b** and **c**, **d**. The line segment $l_{ab}(\tau)$ performs a translation in the direction of the positive x_3–axis or a counterclockwise rotation about the x_3–axis, from time $\tau = 0$ to $\tau = t$.
Wanted: Linear conditions to describe the fact that there is a time τ, $0 < \tau < t$, such that $l_{ab}(\tau)$ and l_{cd} intersect.

In this section we show the following result: For a translational as well as for a rotational motion there exist natural numbers dis, con, dim, so that the following holds:

$$\begin{aligned} &[\exists \tau : \ 0 < \tau < t, \ l_{ab}(\tau) \cap l_{cd} \neq \emptyset] \\ &\iff \bigvee_{i=1}^{dis} \bigwedge_{j=1}^{con} \left[\sum_{k=1}^{dim} \zeta_k^{ij}(\mathbf{c}, \mathbf{d}) \, \delta_k^{ij}(\mathbf{a}, \mathbf{b}, t) \lesseqgtr 0 \right], \end{aligned}$$

where $\zeta_k^{ij}(\mathbf{c}, \mathbf{d})$ is a polynomial in the coordinates of **c** and **d** and $\delta_k^{ij}(\mathbf{a}, \mathbf{b}, t)$ is a polynomial in t and the coordinates of **a** and **b**. These polynomials depend on the kind of motion.

Let L_{ab} and L_{cd} be the lines that contain the segments l_{ab} and l_{cd} respectively. Let $T = \{\tau \,|\, L_{ab}(\tau) \cap L_{cd} \neq \emptyset\}$. Then

$$\begin{aligned} &[\exists \tau \ : \ 0 < \tau < t, \ l_{ab}(\tau) \cap l_{cd} \neq \emptyset] \\ &\quad \iff \ [\exists \tau \in T : \ 0 < \tau < t \ \wedge \ l_{ab}(\tau) \cap l_{cd} \neq \emptyset]. \end{aligned}$$

4.1 Plücker coordinates for lines in $\mathbb{R}^3$

If $L_{ab} \cap L_{cd} \neq \emptyset$, then all four points $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ lie in a plane. In homogeneous coordinates this fact can be expressed by the equation:

$$\det \begin{bmatrix} a_0 & a_1 & a_2 & a_3 \\ b_0 & b_1 & b_2 & b_3 \\ c_0 & c_1 & c_2 & c_3 \\ d_0 & d_1 & d_2 & d_3 \end{bmatrix} = 0$$

Expansion of this 4×4 determinant according to the 2×2 minors of the submatrix formed by the coordinates of the points **a** and **b** and the minors of the submatrix formed by the points **c** and **d** yields the following homogeneous equation:

$$\begin{aligned} 0 = \ &\gamma_{23}\alpha_{01} + \gamma_{31}\alpha_{02} + \gamma_{12}\alpha_{03} \\ &+\gamma_{03}\alpha_{12} + \gamma_{01}\alpha_{23} + \gamma_{02}\alpha_{31} \end{aligned} \tag{2}$$

with $\alpha_{ij} = a_i b_j - a_j b_i$ and $\gamma_{ij} = c_i d_j - c_j d_i$. For the sequel it is convenient to assume that our lines are oriented from the lower to the higher end point, i.e. $a_3 \leq b_3$ and $c_3 \leq d_3$ and hence $\alpha_{03} \geq 0$ and $\gamma_{03} \geq 0$. Moreover we restrict ourselves to the case $\alpha_{03} > 0$ and $\gamma_{03} > 0$, the other cases being simpler.

The Plücker coordinates α_{ij} (and the Plücker coefficients γ_{ij}) are not independent. They fulfill the equations

$$\begin{aligned} \alpha_{01}\,\alpha_{23} + \alpha_{02}\,\alpha_{31} + \alpha_{03}\,\alpha_{12} &= 0, \\ \gamma_{01}\,\gamma_{23} + \gamma_{02}\,\gamma_{31} + \gamma_{03}\,\gamma_{12} &= 0. \end{aligned} \tag{3}$$

With the help of the bilinear equation (2) one can interpret the collision of the two lines L_{ab} and L_{cd} in $\mathbb{R}^3$ as a collision of a point $\mathbf{p}_{ab}$ with a hyperplane H_{cd} in $\mathbb{R}^6$, where $\mathbf{p}_{ab}$ and H_{cd} are given by:

$$\begin{aligned} H_{cd} \ &: \ \gamma_{23}\xi_1 + \gamma_{31}\xi_2 + \gamma_{12}\xi_3 \\ &\quad\ +\gamma_{03}\xi_4 + \gamma_{01}\xi_5 + \gamma_{02}\xi_6 = 0 \\ \mathbf{p}_{ab} \ &: \ (\xi_1, \xi_2, \xi_3, \xi_4, \xi_5, \xi_6)^T = \\ &\quad\ (\alpha_{01}, \alpha_{02}, \alpha_{03}, \alpha_{12}, \alpha_{23}, \alpha_{31})^T \end{aligned}$$

4.2 Collision times for translationally moving lines

In this subsection we compute the possible times of a collision between a translationally moving line L_{ab} and a stationary line L_{cd}.

The translation of the line $L_{ab}(\tau)$ appears in Plücker space as a corresponding motion of the point $\mathbf{p}_{ab}(\tau)$. Its Plücker coordinates are obtained as the 2×2 minors of the following matrix:

$$\begin{bmatrix} a_0 & a_1 & a_2 & a_3+\tau a_0 \\ b_0 & b_1 & b_2 & b_3+\tau b_0 \end{bmatrix},$$

$$\mathbf{p}_{ab}(\tau) = (\alpha_{01}, \alpha_{02}, \alpha_{03}, \alpha_{12}, \alpha_{23} - \tau\,\alpha_{02}, \alpha_{31} + \tau\,\alpha_{01})$$

Substituting these coordinates in the plane equation H_{cd} we obtain:

$$\begin{aligned} & u_1\tau + u_0 = 0 \quad \text{where} \\ u_1 &= \gamma_{02}\alpha_{01} - \gamma_{01}\alpha_{02} \\ u_0 &= \gamma_{23}\alpha_{01} + \gamma_{31}\alpha_{02} + \gamma_{12}\alpha_{03} \\ & \quad + \gamma_{03}\alpha_{12} + \gamma_{01}\alpha_{23} + \gamma_{02}\alpha_{31}. \end{aligned}$$

In the general case, when the projections $\overline{L}_{ab}$ and $\overline{L}_{cd}$ of the lines onto the x_1x_2-plane are not parallel, we get $u_1 \neq 0$ and therefore

$$\tau_0 = -\frac{u_0}{u_1}. \tag{4}$$

Otherwise, if $u_1 = 0$, a collision can only occur if $u_0 = 0$ and $\overline{L}_{ab} = \overline{L}_{cd}$. These conditions describe the following situation: the lines L_{ab} and L_{cd} have to be parallel or to intersect; additionally they must lie in the same plane perpendicular to the x_1x_2-plane. In this case the collision detection of the line segments can be described as a two-dimensional problem.

It is easy to see that a collision between two segments in 2-space is always a collision between a vertex of one segment and the other segment. Appropriate case decompositions yield a linearization of dimension less than 5.

As far as the collision test for polyhedra is concerned these cases can be ignored, because they are detected during the collison test of vertices and facets (see Section 6).

4.3 Conditions for the collision of translationally moving lines

We want to derive linear expressions, which only depend on the coordinates of $\mathbf{a}$ and $\mathbf{b}$, for the predicate $[0 < \tau_0 < t]$. We have the equivalence

$$\begin{aligned} & [0 < \tau_0 < t] \iff \\ & \qquad [u_1 > 0] \;\wedge\; [u_0 < 0] \;\wedge\; [t\,u_1 + u_0 > 0] \\ & \quad \vee \; [u_1 < 0] \;\wedge\; [u_0 > 0] \;\wedge\; [t\,u_1 + u_0 < 0] \end{aligned}$$

where the term $tu_1 + u_0$ can be written in linearized form as follows:

$$\begin{aligned} tu_1 + u_0 &= \gamma_{02}(t\alpha_{01} + \alpha_{31}) + \gamma_{01}(-t\alpha_{02} + \alpha_{23}) \\ & \quad + \gamma_{03}\alpha_{12} + \gamma_{12}\alpha_{03} + \gamma_{23}\alpha_{01} + \gamma_{31}\alpha_{02}. \end{aligned}$$

This gives a linearized form of dimension 6 for the predicate $[0 < \tau_0 < t]$.

4.4 Conditions for the collision of translationally moving line segments

We use the following relation in order to answer the question, whether the line segments really intersect, in case the corresponding lines collide:

$$[l_{ab}(\tau_0) \cap l_{cd} \neq \emptyset] \iff [\bar{l}_{ab}(\tau_0) \cap \bar{l}_{cd} \neq \emptyset]$$

With $\bar{l}_{ab}(\tau_0)$ and $\bar{l}_{cd}$ we denote the projection of the two line segments onto the x_1x_2–plane. Note that $\bar{l}_{ab}(\tau_0) = \bar{l}_{ab}$ because l_{ab} is moving in the positive x_3-direction.

Projection of the line segments onto the x_1x_2–plane

We project the line segments l_{ab} and l_{cd} onto the x_1x_2–plane.

$$\begin{aligned} \bar{l}_{ab}: &\quad \overline{\mathbf{x}} = \overline{\mathbf{a}} + \lambda(\overline{\mathbf{b}} - \overline{\mathbf{a}}), \text{ where } 0 < \lambda < 1 \\ \bar{l}_{cd}: &\quad \overline{\mathbf{x}} = \overline{\mathbf{c}} + \mu(\overline{\mathbf{d}} - \overline{\mathbf{c}}), \text{ where } 0 < \mu < 1 \end{aligned}$$

Then

$$\begin{aligned} & [\bar{l}_{ab} \cap \bar{l}_{cd} \neq \emptyset] \iff \\ & \qquad ([\overline{\mathbf{c}} \text{ left of } \overline{L}_{ab}] \wedge [\overline{\mathbf{d}} \text{ right of } \overline{L}_{ab}] \\ & \qquad \wedge [\overline{\mathbf{a}} \text{ right of } \overline{L}_{cd}] \wedge [\overline{\mathbf{b}} \text{ left of } \overline{L}_{cd}]) \\ & \quad \vee \\ & \qquad ([\overline{\mathbf{c}} \text{ right of } \overline{L}_{ab}] \wedge [\overline{\mathbf{d}} \text{ left of } \overline{L}_{ab}] \\ & \qquad \wedge [\overline{\mathbf{a}} \text{ left of } \overline{L}_{cd}] \wedge [\overline{\mathbf{b}} \text{ right of } \overline{L}_{cd}]). \end{aligned}$$

The point $\overline{\mathbf{c}}$ lies to the left/right of the orientated line $\overline{L}_{ab}$ iff the following is true:

$$\begin{aligned} & ((\mathbf{b} - \mathbf{a}) \times (\mathbf{c} - \mathbf{a}))_3 \lesseqgtr 0 \\ & \iff (\mathbf{a} \times \mathbf{b})_3 + (\mathbf{b} \times \mathbf{c})_3 + (\mathbf{c} \times \mathbf{a})_3 \lesseqgtr 0 \\ & \iff a_1b_2 - a_2b_1 + c_2b_1 - c_1b_2 + c_1a_2 - c_2a_1 \lesseqgtr 0. \end{aligned}$$

Therefore

$$\begin{aligned} & [\bar{l}_{ab} \cap \bar{l}_{cd} \neq \emptyset] \iff \\ & (\;[a_1b_2 - a_2b_1 + c_2b_1 - c_1b_2 + c_1a_2 - c_2a_1 > 0] \\ & \wedge [a_1b_2 - a_2b_1 + d_2b_1 - d_1b_2 + d_1a_2 - d_2a_1 < 0] \\ & \wedge [c_1d_2 - c_2d_1 + d_1a_2 - d_2a_1 + c_2a_1 - c_1a_2 < 0] \\ & \wedge [c_1d_2 - c_2d_1 + d_1b_2 - d_2b_1 + c_2b_1 - c_1b_2 > 0]) \\ & \vee \\ & (\;[a_1b_2 - a_2b_1 + c_2b_1 - c_1b_2 + c_1a_2 - c_2a_1 < 0] \\ & \wedge [a_1b_2 - a_2b_1 + d_2b_1 - d_1b_2 + d_1a_2 - d_2a_1 > 0] \\ & \wedge [c_1d_2 - c_2d_1 + d_1a_2 - d_2a_1 + c_2a_1 - c_1a_2 > 0] \\ & \wedge [c_1d_2 - c_2d_1 + d_1b_2 - d_2b_1 + c_2b_1 - c_1b_2 < 0]). \end{aligned}$$

4.5 Collision times for rotating lines

A counterclockwise rotation of the line $L_{ab}(\varphi)$ about the x_3–axis induces a corresponding motion of the point $\mathbf{p}_{ab}(\varphi)$ in Plücker space. Its Plücker coordinates are given by the 2×2 minors of the following matrix:

$$\begin{bmatrix} a_0 & \cos\varphi\, a_1 + \sin\varphi\, a_2 & -\sin\varphi\, a_1 + \cos\varphi\, a_2 & a_3 \\ b_0 & \cos\varphi\, b_1 + \sin\varphi\, b_2 & -\sin\varphi\, b_1 + \cos\varphi\, b_2 & b_3 \end{bmatrix}$$

$$\mathbf{p}_{ab}(\varphi) = (\cos\varphi\alpha_{01} + \sin\varphi\alpha_{02}, -\sin\varphi\alpha_{01} + \cos\varphi\alpha_{02}, \\ \alpha_{03}, \alpha_{12}, \cos\varphi\alpha_{23} + \sin\varphi\alpha_{31}, -\sin\varphi\alpha_{23} + \cos\varphi\alpha_{31})$$

Substituting these coordinates into the plane equation H_{cd} results in:

$$u_2 \cos\varphi + u_1 \sin\varphi + u_0 = 0 \quad \text{where} \tag{5}$$

$$\begin{aligned} u_2 &= \gamma_{23}\,\alpha_{01} + \gamma_{31}\,\alpha_{02} + \gamma_{01}\,\alpha_{23} + \gamma_{02}\,\alpha_{31} \\ u_1 &= -\gamma_{31}\,\alpha_{01} + \gamma_{23}\,\alpha_{02} - \gamma_{02}\,\alpha_{23} + \gamma_{01}\,\alpha_{31} \\ u_0 &= \gamma_{12}\,\alpha_{03} + \gamma_{03}\,\alpha_{12} \end{aligned}$$

The following parametric formulation

$$\sin\varphi = \frac{2\tau}{1+\tau^2},\ \cos\varphi = \frac{1-\tau^2}{1+\tau^2},\ \text{where } \tau = \tan\frac{\varphi}{2},$$

$0 < \varphi < \pi$, transforms equation (5) into a quadratic equation

$$u_2'\,\tau^2 + u_1'\,\tau + u_0' = 0, \ \text{ where} \tag{6}$$

$$u_2' = u_0 - u_2, \ \ u_1' = 2u_1, \ \ u_0' = u_0 + u_2$$

with the two roots:

$$\tau_1 = \frac{-u_1' + \sqrt{u_1'^2 - 4u_2'u_0'}}{2u_2'}, \ \tau_2 = \frac{-u_1' - \sqrt{u_1'^2 - 4u_2'u_0'}}{2u_2'}$$

for $u_2' \neq 0$. As expected there are in general two points in time where the two lines intersect. If $u_2' = 0$ and $u_1' \neq 0$ there is one collision at time $-\frac{u_0'}{u_1'}$ and an other at ∞, which corresponds to a rotation angle of $\varphi = \pi$. The degenerate case $u_2' = u_1' = 0$ means that a collision during the rotation can only occur if $u_0' = 0$ and the lines lie on the same cone respectively cylinder for which the x_3-axis is the axis of symmetry. Therefore collision detection can be reduced to a 2-dimensional problem, so that we only need to test a collision between vertices and segments.

4.6 Conditions for the collision of rotating lines

τ_1 and τ_2 are real numbers only if $[u_1'^2 - 4u_2'u_0' \geq 0]$. Under this precondition the predicates $[0 < \tau_i < t]$ can be transformed as follows

$$\begin{aligned} &[0 < \tau_1 < t] \Longleftrightarrow \\ &\quad [u_2' > 0] \wedge ([u_1' < 0] \vee [u_0' < 0]) \wedge \ [2tu_2' + u_1' > 0] \\ &\qquad \wedge [t^2 u_2' + tu_1' + u_0' > 0] \\ &\vee [u_2' < 0] \wedge [u_1' > 0] \ \wedge [u_0' < 0] \ \wedge ([2tu_2' + u_1' < 0] \\ &\qquad \vee [t^2 u_2' + tu_1' + u_0' > 0]) \\ &\vee [u_2' = 0] \wedge [u_1' > 0] \ \wedge [u_0' < 0] \ \wedge \ [tu_1' + u_0' > 0]. \end{aligned}$$

For $[0 < \tau_2 < t]$ we get similar conditions. We now derive the linear expressions for the various predicates. From equation (6) we obtain the following equations

$$\begin{aligned} &2tu_2' + u_1' = \\ &\quad 2\gamma_{03}(t\alpha_{12}) + 2\gamma_{12}(t\alpha_{03}) + 2\gamma_{23}(-t\alpha_{01} + \alpha_{02}) \\ &\quad +2\gamma_{01}(-t\alpha_{23} + \alpha_{31}) + 2\gamma_{02}(-t\alpha_{31} - \alpha_{23}) \\ &\quad +2\gamma_{31}(-t\alpha_{02} - \alpha_{01}) \end{aligned}$$

$$\begin{aligned} &t^2 u_2' + tu_1' + u_0' = \\ &\quad \gamma_{02}(-t^2\alpha_{31} - 2t\alpha_{23} + \alpha_{31}) + \gamma_{12}(t^2\alpha_{03} + \alpha_{03}) \\ &\quad +\gamma_{01}(-t^2\alpha_{23} + 2t\alpha_{31} + \alpha_{23}) + \gamma_{03}(t^2\alpha_{12} + \alpha_{12}) \\ &\quad +\gamma_{23}(-t^2\alpha_{01} + 2t\alpha_{02} - \alpha_{01}) \\ &\quad +\gamma_{31}(-t^2\alpha_{02} - 2t\alpha_{01} + \alpha_{02}), \end{aligned}$$

$$\begin{aligned} &u_1'^2 - 4u_2'u_0' = \\ &\quad (\gamma_{23}^2 + \gamma_{31}^2)(\alpha_{01}^2 + \alpha_{02}^2) + (\gamma_{01}^2 + \gamma_{02}^2)(\alpha_{23}^2 + \alpha_{31}^2) \\ &\quad -2(\gamma_{02}\gamma_{23} - \gamma_{01}\gamma_{31})(\alpha_{02}\alpha_{23} - \alpha_{01}\alpha_{31}) \\ &\quad -\gamma_{12}^2(\alpha_{03}^2) - \gamma_{03}^2(\alpha_{12}^2). \end{aligned} \tag{7}$$

In each case the predicates $[t^2u_2' + tu_1' + u_0' \lesseqgtr 0]$, $[2tu_2' + u_1' \lesseqgtr 0]$ and $[u_1'^2 - 4u_2'u_0' \geq 0]$ are linear in at most 6 expressions $\delta_k^{ij}(\mathbf{a}, \mathbf{b}, t)$, which are given as polynomials in the coordinates of $\mathbf{a}$ and $\mathbf{b}$ and the time parameter t. This means we have found a linearization of dimension 6 for the predicate $[0 < \tau_i < t]$.

4.7 Conditions for the collision of rotating line segments

We reduce the decision whether $[l_{ab}(\tau_i) \cap l_{cd} \neq \emptyset]$ (in case the corresponding lines collide) to the calculation of the x_3–coordinate z of the intersection points of the corresponding lines and a test whether $z \in [\min\{a_3, b_3\}, \max\{a_3, b_3\}] \cap [\min\{c_3, d_3\}, \max\{c_3, d_3\}]$. For the calculation of the x_3–coordinates of the possible intersection points we use cylindrical coordinates.

Representation of line segments in cylindrical coordinates

If we represent the line segments

$$\begin{aligned} l_{ab}: &\quad \mathbf{x} = \mathbf{a} + \lambda(\mathbf{b} - \mathbf{a}), \quad \text{where} \quad 0 \leq \lambda \leq 1, \\ l_{cd}: &\quad \mathbf{x} = \mathbf{c} + \mu(\mathbf{d} - \mathbf{c}), \quad \text{where} \quad 0 \leq \mu \leq 1, \end{aligned}$$

in cylindrical coordinates (r, φ, z), we can easily check, whether the line segment l_{ab} can collide with the line segment l_{cd} during a full rotation about the x_3–axis. During its rotation the line segment l_{ab} generally describes a hyperboloid, whose projection onto the (r, z)–plane of the cylindrical coordinate system yields a hyperbolic segment. The rotating line segment l_{ab} can only collide with l_{cd}, if the two corresponding hyperbolic segments intersect in the (r, z)–plane. In order to compute this intersections we have to find the (r, z)-representation for each point in $\mathbb{R}^3$, which is given by its Cartesian coordinates (x_1, x_2, x_3). We get

$$z = x_3 \ , \ r = \sqrt{x_1^2 + x_2^2}.$$

Since $x_i = a_i + \lambda(b_i - a_i)$ and $\lambda = \dfrac{z - a_3}{b_3 - a_3}$, we have for $a_3 \neq b_3$:

$$r^2 = \frac{1}{(b_3 - a_3)^2}\big((a_1 b_3 - a_3 b_1 + z(b_1 - a_1))^2 + (a_2 b_3 - a_3 b_2 + z(b_2 - a_2))^2\big).$$

We proceed with Plücker coordinates and get:

$$\begin{aligned} r^2 &= \frac{1}{\alpha_{03}^2}\left((\alpha_{31} - z\alpha_{01})^2 + (\alpha_{23} + z\alpha_{02})^2\right) \\ &= v_2 z^2 + v_1 z + v_0, \end{aligned}$$

where $v_2 = \frac{\alpha_{01}^2 + \alpha_{02}^2}{\alpha_{03}^2}$, $v_1 = 2\frac{\alpha_{02}\alpha_{23} - \alpha_{01}\alpha_{31}}{\alpha_{03}^2}$, and $v_0 = \frac{\alpha_{23}^2 + \alpha_{31}^2}{\alpha_{03}^2}$. The question whether the line segment l_{ab} collides with the stationary segment l_{cd} while it is rotating about the x_3–axis can be answered by calculating the intersection between the following two parabolic segments:

$$\begin{aligned} r_{ab}^2(z) &= v_2 z^2 + v_1 z + v_0 \text{ with } a_3 \leq z \leq b_3, \\ r_{cd}^2(z) &= w_2 z^2 + w_1 z + w_0 \text{ with } c_3 \leq z \leq d_3. \end{aligned}$$

W.l.o.g. let the line segments be given, such that $a_3 \leq b_3$ and $c_3 \leq d_3$. The intersection points of the two parabola can be found as the roots of a quadratic equation:

$$v_2' z^2 + v_1' z + v_0' = 0 \quad \text{where} \quad v_i' = v_i - w_i, \tag{8}$$

$$z_1 = \frac{-v_1' + \sqrt{v_1'^2 - 4v_2' v_0'}}{2v_2'}, \ z_2 = \frac{-v_1' - \sqrt{v_1'^2 - 4v_2' v_0'}}{2v_2'},$$

for $v_2' \neq 0$.

But a collision of the rotating line segment l_{ab} with l_{cd} exists only if the quadratic equation has real roots ($[v_1'^2 - 4v_2' v_0' \geq 0]$) and these lie in the interval $[a_3, b_3] \cap [c_3, d_3]$. For $v_2' = 0$ and $v_1' \neq 0$ we get one solution at $-\frac{v_0'}{v_1'}$ and an other at ∞. The case $v_2' = v_1' = v_0' = 0$ occurs iff L_{cd} lies on the hyperboloid generated by the rotating line L_{ab}. In this case possible collisions can be ignored because they are detected when testing vertices against facets.

By using cylindrical coordinates we have succeeded in finding the x_3–coordinates of the possible intersection points of the rotating line $L_{ab}(\tau)$ with L_{cd}. But it remains open, to which x_3–coordinate the collision time τ_i corresponds. It can be shown that z_1 belongs to τ_1 and z_2 to τ_2, if we assume that $a_3 < b_3$ and $c_3 < d_3$. That is

$$\begin{aligned} &[l_{ab}(\tau_i) \cap l_{cd} \neq \emptyset] \Longleftrightarrow \\ &\qquad [z_i > a_3] \wedge [z_i < b_3] \wedge [z_i > c_3] \wedge [z_i < d_3]. \end{aligned} \tag{9}$$

In the following we want to derive linearized conditions for the predicates $[z_i < Z]$ respectively $[z_i > Z]$ For example we get

$$\begin{aligned} &[z_1 < Z] \Longleftrightarrow \\ &\quad [v_2' > 0] \wedge [2Zv_2' + v_1' > 0] \wedge [Z^2 v_2' + Z v_1' + v_0' > 0] \\ &\vee [v_2' < 0] \wedge ([2Zv_2' + v_1' < 0] \vee [Z^2 v_2' + Z v_1' + v_0' > 0]) \\ &\vee [v_2' = 0] \wedge \qquad [v_1' > 0] \wedge \qquad [Z v_1' + v_0' > 0] \end{aligned}$$

the other cases being similar. It holds:

$$\begin{aligned} v_2' &= \frac{\alpha_{01}^2 + \alpha_{02}^2}{\alpha_{03}^2} - \frac{\gamma_{01}^2 + \gamma_{02}^2}{\gamma_{03}^2}, \\ v_1' &= 2\frac{\alpha_{02}\alpha_{23} - \alpha_{01}\alpha_{31}}{\alpha_{03}^2} - 2\frac{\gamma_{02}\gamma_{23} - \gamma_{01}\gamma_{31}}{\gamma_{03}^2}, \\ v_0' &= \frac{\alpha_{23}^2 + \alpha_{31}^2}{\alpha_{03}^2} - \frac{\gamma_{23}^2 + \gamma_{31}^2}{\gamma_{03}^2}. \end{aligned} \tag{10}$$

The predicates $[v_1'^2 - 4v_2' v_0' \geq 0]$ and $[u_1'^2 - 4u_2' u_0' \geq 0]$ are equivalent, since both express the fact that the rotating line L_{ab} collides with the stationary line L_{cd} during a full rotation. On the basis of relations (10) and equation (3) it holds:

$$u_1'^2 - 4u_2' u_0' \;=\; \alpha_{03}^2 \gamma_{03}^2 \cdot (v_1'^2 - 4v_2' v_0'). \tag{11}$$

According to equation (7) the predicate $[v_1'^2 - 4v_2' v_0' \geq 0]$ is linear in the expressions $\alpha_{01}^2 + \alpha_{02}^2$, $\alpha_{23}^2 + \alpha_{31}^2$, $\alpha_{01}\alpha_{31} - \alpha_{02}\alpha_{23}$, α_{03}^2 and α_{12}^2.

Now let us consider the predicates $[v_2' \lesseqgtr 0]$, $[2Zv_2' + v_1' \lesseqgtr 0]$ and $[Z^2 v_2' + Z v_1' + v_0' \lesseqgtr 0]$. A multiplication with $\alpha_{03}^2 \gamma_{03}^2$ yields:

$$\begin{aligned} &[v_2' \lesseqgtr 0] \Longleftrightarrow \\ &\quad \big[\gamma_{03}^2(\alpha_{01}^2 + \alpha_{02}^2) - (\gamma_{01}^2 + \gamma_{02}^2)(\alpha_{03}^2) \lesseqgtr 0\big], \end{aligned}$$

$$\begin{aligned} &[2Zv_2' + v_1' \lesseqgtr 0] \Longleftrightarrow \\ &\quad \big[2Z\gamma_{03}^2(\alpha_{01}^2 + \alpha_{02}^2) - 2Z(\gamma_{01}^2 + \gamma_{02}^2)(\alpha_{03}^2) \\ &\quad + 2\gamma_{03}^2(\alpha_{02}\alpha_{23} - \alpha_{01}\alpha_{31}) \\ &\quad - 2(\gamma_{02}\gamma_{23} - \gamma_{01}\gamma_{31})(\alpha_{03}^2) \lesseqgtr 0\big], \end{aligned}$$

$$\begin{aligned} &[Z^2 v_2' + Z v_1' + v_0' \lesseqgtr 0] \Longleftrightarrow \\ &\quad \big[Z^2\gamma_{03}^2(\alpha_{01}^2 + \alpha_{02}^2) - Z^2(\gamma_{01}^2 + \gamma_{02}^2)(\alpha_{03}^2) \\ &\quad + 2Z\gamma_{03}^2(\alpha_{02}\alpha_{23} - \alpha_{01}\alpha_{31}) - (\gamma_{23}^2 + \gamma_{31}^2)(\alpha_{03}^2) \\ &\quad - 2Z(\gamma_{02}\gamma_{23} - \gamma_{01}\gamma_{31})(\alpha_{03}^2) + \gamma_{03}^2(\alpha_{23}^2 + \alpha_{31}^2) \lesseqgtr 0\big]. \end{aligned}$$

If $Z = c_3, d_3$ (see condition 10), then these predicates are linear in the four expressions $\alpha_{01}^2 + \alpha_{02}^2$, α_{03}^2, $\alpha_{02}\alpha_{23} - \alpha_{01}\alpha_{31}$, $\alpha_{23}^2 + \alpha_{31}^2$, and the corresponding coefficients only depend on the coordinates of the points **c** and **d**. However if $Z = a_3, b_3$, then one can define the expressions $Z^k\,(\alpha_{01}^2 + \alpha_{02}^2)$, $Z^k\,\alpha_{03}^2$, $Z^k\,(\alpha_{02}\alpha_{23} - \alpha_{01}\alpha_{31})$, $Z^k\,(\alpha_{23}^2+\alpha_{31}^2)$ for $k = 0, 1, 2$, so that the former predicates are linear in at most six of these expressions, where again the corresponding coefficients only depend on the coordinates of the points **c** and **d**.

4.8 Linearization

To summarize we firstly computed the conditions for the fact that the moving line L_{ab} intersects the stationary line L_{cd} during a time interval $[0, t]$. In the next step (see Sections 4.4 and 4.7) we got the additional conditions for the intersection of the corresponding line segments. The combination of these two sets of conditions gives the wanted linearization, i.e.

$$[\exists \tau \;:\; 0 < \tau < t,\; l_{ab}(\tau) \cap l_{cd} \neq \emptyset] \iff \bigvee_i ([0 < \tau_i < t] \wedge [l_{ab}(\tau_i) \cap l_{cd} \neq \emptyset])$$

Until now we have found linearizations with $dim = 6$ where $\delta_k^{ij}(\mathbf{c}, \mathbf{d})$ are polynomials in the coordinates of **c** and **d** and $\zeta_k^{ij}(\mathbf{a}, \mathbf{b}, t)$ are polynomials in t and the coordinates of **a** and **b**. In order to reduce the dimension we divide each inequality by a positive coefficient $\zeta_k^{ij}(\mathbf{a}, \mathbf{b}, t)$, which we can always find using the fact that $\alpha_{03} > 0$. So we get a linearization with dimension $dim = 5$. For example let us consider the condition $[2tu_2' + u_1' > 0]$ (see (7)).

We can divide the inequality by the term $2t\alpha_{03} > 0$ and replace the inequality $2tu_2' + u_1' > 0$ by

$$\begin{aligned} 0 < \gamma_{01}\frac{-t\alpha_{23} + \alpha_{31}}{t\alpha_{03}} &+ \gamma_{02}\frac{-t\alpha_{31} - \alpha_{23}}{t\alpha_{03}} \\ +\gamma_{03}\frac{t\alpha_{12}}{t\alpha_{03}} &+ \gamma_{23}\frac{-t\alpha_{01} + \alpha_{02}}{t\alpha_{03}} \\ +\gamma_{31}\frac{-t\alpha_{02} - \alpha_{01}}{t\alpha_{03}} &+ \gamma_{12}. \end{aligned}$$

Now we can apply the construction of Section 3.2, especially Corollary 4.

5 Improved solution for the translational case

Applying a recent result of Pellegrini we can construct a faster algorithm for translationally moving line segments. During its movement a segment moves over a quadrilateral which all segments, that are hit during the translation, have to intersect. We can triangulate the quadrilateral and determine all segments intersecting one of the resulting triangles. During this process it can happen that we count segments twice, if they intersect the common edge of the two triangles. But for our emptiness problem it is sufficient to work inaccurately. We can count the number of collisions and when there are less than three of them we compute them and check whether they are different. Therefore we have to consider the following subproblem:

> Given a set L of n line segments, construct a data structure such that for any query triangle one can efficiently count/compute the incidences with L.

In [Pel92] and [AM92] there is a solution based on Plücker coordinates of lines. In contrast to our method based on Theorem 1, the construction of the data structures for points and hyperplanes in Plücker space makes use of the result that the zone of the Plücker hypersurface among n hyperplanes has size $O(n^4 \log n)$ ([APS93]). This additional structure is essential to obtain a final bound of $O(n^{8/5+\epsilon})$.

Theorem 6 *Given n segments there is a data structure using $O(m)$ space, $n^{1+\epsilon} \leq m \leq n^{4+\epsilon}$, such that we can count the number of segments intersected by a query triangle in time $O(n^{1+\epsilon}/m^{1/4})$. For reporting these k segments we need $O(k)$ additional steps. The structure can be build in time $O(m)$.*

The query algorithm can be replaced by a parallel version running in $O(\text{polylog } n)$ parallel steps using $O(n^{1+\epsilon}/m^{1/4})$ processors, which allows us to apply the *parametric search technique* efficiently.

Corollary 7 *Given a set of n line segments, each of which is moving translationally in some fixed direction over distance t and a set of n stationary line segments in $\mathbb{R}^3$ we can compute the first collision between the two sets in time $O(n^{8/5+\epsilon})$.*

6 Collisions between facets and vertices

Recall that we consider two polyhedra one of which is moving (let us say P_1) whereas the other one, P_2, is stationary. Let V_i, E_i, F_i, $i = 1, 2$, denote the sets of vertices, edges and facets of the polyhedra. Only translations or rotations are permitted as motions. Until now we have shown how to compute the first collision between the edges of P_1 and P_2. We still have to determine the first facet of P_2 hit by a vertex of P_1 respectively the first facet of the moving polyhedron which collides with a vertex of P_2. A solution to this problem, which applies to both kinds of motions, is already presented in [Sch94] based on ideas from [Nu85]. The facets and vertices are projected onto a 2-dimensional space and a plane sweep technique is applied. For completeness we present a sketch of this construction. We only determine the time of the first collision between the set of vertices of P_1 and the set of facets of P_2.

In case of a translation we project the facets and vertices onto a plane perpendicular to the direction of the motion. If the polyhedron P_1 is rotating about an axis (see Section 4.7) we can apply a similar method, but we have to work with cylindrical coordinates: the projection is done by removing the angle-component. In both cases we get in the projection-plane a point set $\overline{V_1}$ which is the image of the vertices of P_1 and 2-dimensional regions $\overline{F_2}$ bounded by line segments respectively hyperbola segments. Now we execute a plane sweep: stop-points/halts are starting- and end-points, extremal- and intersection-points of the segments. Between two consecutive halts the ordering of the intersection-points between the segments and the plane sweep S is always the same. Therefore we can save the active segments in a balanced search tree which will be the primary structure for saving more informations.

Let R be a region between two segments, which are adjacent on S, and assume that S goes over R between two consecutive halts. Every vertex of P_1, whose projection $\overline{v}$ lies in R, can only collide with facets of P_2, the projections $\overline{f}$ of which contain R. Therefore for each region R we keep track of the set $F_R = \{f \in F_2 | R \subseteq \overline{f}\}$. Thus for every segment we save the set F_R of the region lying above it.

Dependent on the kind of motion we can define an ordering for each set F_R. During a rotation every point with projection in R will stab the regions in F_R with the same cyclic ordering; in the case of a translation we will get a linear order. As secondary structure which stores the elements in F_R we again use a balanced search tree which allows to find the first facet hit by a vertex projected onto R in logarithmic time.

The sweep line stops at every point $\overline{v} \in \overline{V}$. There we determine the region R containing $\overline{v}$ and search the facet of F_R which is hit by the corresponding vertex first. Both steps can be done in logarithmic time using the tree structure.

During the sweep we have to hold all regions intersected by the sweep plane S as well as the sets F_R. To save space we only store the changes of the sets F_R (see [Nu85]). Using this idea each set F_R can be stored with logarithmical costs.

The run time of the algorithm is $O((|V_1| + |E_2| + C_{\overline{E_2}}) \log |E_2|)$ where $C_{\overline{E_2}}$ denotes the number of intersections of the projected edges of P_2. Unfortunately this value could be quadratic in the complexity of the polyhedron. Therefore we divide the problem into several smaller subproblems. W.l.o.g. we assume that the facets of P_2 are triangles (a triangulation of the surface does not change the asymptotic complexity of P_2). We divide F_2 in $\sqrt{|F_2|}$ many subsets of size $\sqrt{|F_2|}$. For each subset we execute the above plane sweep algorithm.

Theorem 8 *Consider two polyhedra P_1, P_2, one of which is moving translationally or rotating about a fixed axis. The first collision between a vertex of one of them and a facet of the other can be computed in time $O((|P_1| + |P_2|)^{3/2} \log(|P_1| + |P_2|))$.*

As already in the case of edge/edge collisions there is an improved solution for the translational motion. This follows from the fact that vertical ray shooting into a set of disjoint triangles can be done with $O(n^{4/3+\epsilon})$ preprocessing and $O(n^{1/3})$ query time. The algorithm for this kind of ray shooting selects all triangles intersected by the line containing the query ray and determines the first triangle that is hit using binary search. We show how this can be done by simplex composition. We project all triangles and the query ray, respectively the supporting line, onto the plane perpendicular to the direction of the motion. A query line intersects a triangle if and only if the projection of the line lies within the projection of the triangle. Since the projection of the line results in a point in the plane the first part of the algorithm corresponds to planar point location which is dual to halfspace range searching in the plane; in the second step we have to find the first triangle hit by the query ray among all the triangles stabbed by the supporting vertical line. This can be done simply by sorting and searching. The projection of the triangles divides the plane into cells each of which corresponds to a certain stabbing order, i.e. all points in such a cell are the images of vertical lines intersecting the same triangles in the same order. The projection of the supporting line has to lie in the common intersection of all stabbed triangles. We compute this region, choose an arbitrary point in it and determine the stabbing order of the corresponding line, which is the same for all points in the region, in time $O(n \log n)$. This finishes the preprocessing. To find the first triangle hit by the query ray we do binary search on this order. Comparisons are made by determining the relative position of the ray's starting point with respect to a triangle. Applying Theorem 1 we get

Theorem 9 *Consider two polyhedra P_1, P_2, one of which is moving translationally. The first collision between a vertex of one of them and a facet of the other can be computed in time $O(n^{4/3+\epsilon})$.*

7 Conclusion

We have shown how to determine the collision between a stationary and a moving polyhedron in sub-quadratic time. For that we have computed the first collision between vertices of one polyhedron and facets of the other and the first collision between the edges of the polyhedra. We have reduced the latter task to the formulation of an appropriate linearization which is derived from an explicit computation of the collision times. We could do this because the equations of the motions have degree at most two. The natural question is how we can proceed if the motion of the polyhedron is more complicated, i.e. if the equations have degree greater than five (then no explicit formulation of the roots exists).

Actually we do not need an explicit representation of the collision times for the linearization. We only need to know whether two particular features of the polyhedra collide during a given time period. Formally we come upon the problem whether a system of (constant many) multivariate polynomial equations with constant degree has a common real solution in a box or not. Applying recent results on counting real zeros of a system of algebraic equations [P91, US92] one can decide this question using a finite number of algebraic operations on the coefficients of the polynomials and the vertices of the box. This extension of Sturm's theorem to arbitrary dimensions is purely symbolic so we can use it to get appropriate linearizations, which leads to subquadratic collision detection algorithms for a wide class of motions.

References

[AM92] P. K. Agarwal, J. Matoušek: *On range searching with semialgebraic sets*, Proc. 17th Symp. on Math. Foundation of CS: 1–13, (1992)

[AM92B] P. K. Agarwal, J. Matoušek: *Ray shooting and parametric search*, Proc. 24th STOC: 517–526, (1992)

[APS93] B. Aronov, M. Pellegrini, and M. Sharir: *On the zone of an algebraic surface in a hyperplane arrangement*, Discrete Comput. Geom 9: 177–188, (1993)

[BO79] J. W. Boyse: *Interference detection among solids and surfaces*, CACM Vol. 22(1): 3–9, (1979)

[CA84] J. Canny: *On detecting collision between polyhedra*, Proc. ECAI (1984), 533–542

[DK85] D. Dobkin, D. Kirkpatrick: *A linear algorithm for determing the seperation of convex polyhedra*, J. Algor. 6 (1985), 381–392

[DK90] D. Dobkin, D. Kirkpatrick: *Determining the seperation of preprocessed polyhedra – a unified approach*, Lecture Notes in Computer Science 443 (1990), 400–413

[DHKS90] D. Dobkin, J. Hershberger, D. Kirkpatrick, S. Suri: *Implicitily searching convolutions and computing depth of collisions*, Lecture Notes in Computer Science 450 (1990), 165–180

[KREV92] M. van Kreveld: *New Results on Data Structures in Computational Geometry*, Ph.D. Thesis, University of Utrecht, The Netherlands, (1992)

[MAT93] J. Matoušek: *Range searching with efficient hierarchical cuttings*, Discrete Comput. Geom 10: 159–182, (1993)

[MEG83] N. Megiddo: *Applying parallel computation algorithms in the design of serial algorithms.* Journal of the ACM **30**: 852–865, (1983).

[NU85] O. Nurmi: *A fast algorithm for hidden-line elimination*, BIT, Vol. 25, 466–472, (1985)

[P91] P. Pedersen: *Counting real zeros*, New York University, NYU Technical Report 545-R243, (1991)

[PEL92] M. Pellegrini: *Incidence and nearest-neighbor problems for lines in* 3*-space*, Proc. 8th Annu. ACM Sympos. Comput. Geom.: 130–137, (1992)

[PEL93] M. Pellegrini: *Ray shooting on triangles in* 3*-space*, Algorithmica 9: 471–494, (1993)

[PS88] F. P. Preparata and M. I. Shamos: *Computational Geometry: an Introduction*, Springer-Verlag, New York, (1988)

[SCH94] E. Schömer: *Interaktive Montagesimulation mit Kollisionserkennung*, Ph.D. Thesis, Universität des Saarlandes, Germany, (1994)

[US92] A. Y. Uteshev and S. G. Shulyak: *Hermite's method of seperation of solutions of systems of algebraic equations and its applications*, Linear Algebra and its Applications 177: 49–88, (1992)

[YY85] A. C. Yao, F. F. Yao: *A general approach to D-dimensional geometric queries*, Proc. 17th Annu. Sympos. Theory Comput.: 163–168, (1985)