Technische Universität Graz

Collision Detection for Complicated Polyhedra
Using the Fast Multipole Method or Ray Crossing

B. Muth, G. Of, P. Eberhard, O. Steinbach

Berichte aus dem
Institut für Numerische Mathematik

Bericht 2006/3

Technische Universität Graz

Collision Detection for Complicated Polyhedra Using the Fast Multipole Method or Ray Crossing

B. Muth, G. Of, P. Eberhard, O. Steinbach

Berichte aus dem
Institut für Numerische Mathematik

Bericht 2006/3

Technische Universität Graz
Institut für Numerische Mathematik
Steyrergasse 30
A 8010 Graz

WWW: `http://www.numerik.math.tu-graz.at`

Collision Detection for Complicated Polyhedra Using the Fast Multipole Method or Ray Crossing

Beate Muth*, Günther Of#, Peter Eberhard*, and Olaf Steinbach#

* Institute of Engineering and Computational Mechanics, University of Stuttgart, Pfaffenwaldring 9, 70569 Stuttgart, Germany,
[muth,eberhard]@itm.uni-stuttgart.de

Institute for Computational Mathematics, Graz University of Technology, Steyrergasse 30, 8010 Graz, Austria,
[of,o.steinbach]@tugraz.at

Abstract The purpose of this paper is a comparison of two different methods that can be used for collision detection. One method is called the ray crossing method, a commonly used geometrical approach. The other method is the fast multipole method, usually used for boundary element methods, which is also applied for collision detection purposes here. Both methods are especially of interest when the collision for arbitrarily shaped polyhedra has to be detected. Here, both methods are described and compared for different examples of complex shaped polyhedra with up to $5 \cdot 10^5$ faces and more than $5 \cdot 10^5$ test points regarding efficiency and required calculation time.

Key words: contact, collision detection, fast multipole method, boundary element method, ray crossing, molecular dynamics, point-in-polygon test

1 Introduction

The background of this study is the modeling and simulation of bulk good. There, as a brief background information and motivation for the following, the calculation of the contact forces that are applied to the bodies due to contacts are often modelled by means of the molecular dynamics (MD) method, where penetrations between bodies appear. Contact forces ensure a physical behavior of the particles.

In order to simulate the dynamical behavior of a system with different bodies and frequently changing contact forces, it is necessary to know precisely which bodies are in contact. In this paper, the efficiency of two very different methods for collision detection of arbitrarily shaped bodies is investigated. For arbitrarily shaped bodies here a frequently applied description is used, i.e. the surfaces of bodies are represented as compounds of polygons and are therefore polyhedra.

For an efficient simulation often collision detection is the bottle neck. Therefore, it is essential to use an efficient method. Originally these methods basically check for every body pair, whether a node of one body is inside the other body. This corresponds to the classical point-in-polygon test from computational geometry. In this study, two different methods for collision detection are compared with respect to efficiency in computation time. In order to compare these methods properly, our investigation foregrounds the efficiency of collision detection methods for a series of bodies consisting of a very large number of geometric primitives, since for systems with only few primitives collision detection is not the critical point.

The two collision detection methods that are used in order to find these colliding body pairs are explained in Section 2. A very unusual approach for collision detection, the fast multipole method, is suggested in Subsection 2.1. There, a double layer potential of a body surface is evaluated and by means of this value the decision can be made whether a point of another body is inside or outside this body. For comparison, in Subsection 2.2 a geometric approach, the ray crossing method for collision detection is explained. In Section 3 both methods are compared using several test examples and, finally some results are compared and discussed with respect to geometrically very complicated applications.

2 Collision detection

As soon as neighboring body pairs are identified using some kind of sorting phase [1, 2, 31], the precise but time consuming collision detection can be carried out. For these body-pairs it is necessary to check, whether and which surface points of the bodies are located inside another body, i.e. the point-in-polyhedron test must be performed many times. Since this procedure is very time consuming, it is obligatory to use an efficient method in order to perform this test. In the following, two very different methods are explained that can be used for the point-in-polygon checks.

2.1 Double layer potential and fast multipole method

The first approach is based on a special property of the double layer potential of the boundary element method (BEM) and the fast multipole method (FMM) providing a fast evaluation of boundary integral operators. The main idea is to compute a certain surface potential in evaluation points, which are here the surface nodes to be checked. Therefore, we want to investigate an evaluation point, denoted by the position vector $\mathbf{x}$, which is tested for collision with a body Y, whose surface is discretized by a boundary mesh consisting of triangles. The two ingredients of the method, the BEM and the FMM, are described briefly in the following subsections and only as far as required for the collision detection.

2.1.1 The double layer potential

Let $Y \subset \mathbb{R}^3$ be a bounded, simply connected domain with Lipschitz boundary $\Gamma = \partial Y$. The position vectors of nodes of this body Y may be denoted as $\mathbf{y}$. For the Laplace equation

$$-\Delta u(\mathbf{y}) = 0 \qquad \text{for all } \mathbf{y} \in Y, \tag{1}$$

the second Green identity

$$-\int\limits_{Y} \Delta u(\mathbf{y}) v(\mathbf{y}) \mathrm{d}\mathbf{y} + \int\limits_{\Gamma} \left[\mathbf{n}_{\mathbf{y}} \cdot \nabla_{\mathbf{y}} u(\mathbf{y})\right] v(\mathbf{y}) \mathrm{d}s_{\mathbf{y}} = \\ -\int\limits_{Y} \Delta v(\mathbf{y}) u(\mathbf{y}) \mathrm{d}\mathbf{y} + \int\limits_{\Gamma} \left[\mathbf{n}_{\mathbf{y}} \cdot \nabla_{\mathbf{y}} v(\mathbf{y})\right] u(\mathbf{y}) \mathrm{d}s_{\mathbf{y}} \tag{2}$$

is valid. Here, $\mathbf{n}_{\mathbf{y}}$ denotes the outer normal vector given for almost all $\mathbf{y} \in \Gamma$, $\int_{\Gamma} \bullet \, \mathrm{d}s_{\mathbf{y}}$ is the integral over the boundary Γ described by triangles, and $v(\mathbf{y})$ is an arbitrary, sufficiently smooth function. With the constant solution $v(\mathbf{y}) = 1$ of the Laplace equation, this formula is reduced to

$$-\int\limits_{Y} \Delta u(\mathbf{y}) \mathrm{d}\mathbf{y} + \int\limits_{\Gamma} \mathbf{n}_{\mathbf{y}} \cdot \nabla_{\mathbf{y}} u(\mathbf{y}) \mathrm{d}s_{\mathbf{y}} = 0. \tag{3}$$

For $\mathbf{x} \neq \mathbf{y}$, the fundamental solution

$$u(\mathbf{y}) := U^*(\mathbf{x}, \mathbf{y}) = \frac{1}{4\pi} \frac{1}{|\mathbf{x} - \mathbf{y}|} \tag{4}$$

of the Laplace operator in Eq. (1) is considered. In order to ensure $\mathbf{x} \neq \mathbf{y}$, which means that $\mathbf{x}$ may not be equal to any point of domain Y, a small neighborhood of $\mathbf{x}$ is cut off from the domain Y by a small sphere $|\mathbf{x} - \mathbf{y}| < \varepsilon$. Then, inserting Eq. (4) into Eq. (3) for the modified domain $Y \setminus \{y \in Y : |\mathbf{x} - \mathbf{y}| < \varepsilon\}$ leads to

$$\begin{aligned} -\frac{1}{4\pi} \int\limits_{\Gamma} \mathbf{n}_{\mathbf{y}} \cdot \nabla_{\mathbf{y}} \frac{1}{|\mathbf{x} - \mathbf{y}|} \mathrm{d}s_{\mathbf{y}} &= \frac{1}{4\pi} \int\limits_{\mathbf{y} \in Y : |\mathbf{x} - \mathbf{y}| = \varepsilon} \mathbf{n}_{\mathbf{y}} \cdot \nabla_{\mathbf{y}} \frac{1}{|\mathbf{x} - \mathbf{y}|} \mathrm{d}s_{\mathbf{y}} \\ &= \frac{1}{4\pi} \frac{1}{\varepsilon^2} \int\limits_{\mathbf{y} \in Y : |\mathbf{x} - \mathbf{y}| = \varepsilon} \mathrm{d}s_{\mathbf{y}}. \end{aligned} \tag{5}$$

For arbitrary test points $\mathbf{x} \in \mathbb{R}^3$, the functional $J : \mathbb{R}^3 \to [0, 1]$ defined as

$$J(\mathbf{x}) := -\frac{1}{4\pi} \int\limits_{\Gamma} \mathbf{n}_{\mathbf{y}} \cdot \nabla_{\mathbf{y}} \frac{1}{|\mathbf{x} - \mathbf{y}|} \mathrm{d}s_{\mathbf{y}} \tag{6}$$

has the property

$$J(\mathbf{x}) = \lim_{\varepsilon \to 0} \frac{1}{4\pi} \frac{1}{\varepsilon^2} \int\limits_{\mathbf{y} \in Y : |\mathbf{x} - \mathbf{y}| = \varepsilon} \mathrm{d}s_{\mathbf{y}}, \tag{7}$$

see Eq. (5), where the right hand side describes some kind of solid angle. Note that the functional J is the double layer potential applied to a function $v(\mathbf{y}) = 1$. In the case of a smooth boundary, the functional $J(\mathbf{x})$ takes only the values, see e.g. [32],

$$J(\mathbf{x}) = \begin{cases} 1 & \text{for } \mathbf{x} \in Y, \\ 0 & \text{for } \mathbf{x} \notin Y, \\ \frac{1}{2} & \text{for } \mathbf{x} \in \Gamma. \end{cases} \tag{8}$$

Therefore, this functional $J(\mathbf{x})$ can be used to decide whether a point $\mathbf{x}$ is inside ($J(\mathbf{x}) = 1$) or outside ($J(\mathbf{x}) = 0$) the domain Y. If $J(\mathbf{x}) \in]0, 1[$ the point $\mathbf{x}$ is on the boundary Γ and the value indicates some kind of angle of the boundary in $\mathbf{x}$.

The potential $J(\mathbf{x})$ can be exactly computed via definition (6) in the case that the boundary of Y is described by plane triangles. A first approach of the method is based on this exact evaluation of the functional. Let the boundary Γ be given by a triangulation using N boundary elements. Then, the functional $J(\mathbf{x})$ has to be computed for each of these N elements in order to know whether a point $\mathbf{x}$ is inside or outside the domain. If X contains M points, the functional $J(\mathbf{x})$ is evaluated in M evaluation points $\mathbf{x}_\ell$. Then, the corresponding effort will be of order $\mathcal{O}(NM)$ as the functional J has to be evaluated for all combinations. The evaluation effort of J is comparable to the effort evaluating the kernel on the discrete level

$$k(\mathbf{x}, \mathbf{y}) := \frac{1}{|\mathbf{x} - \mathbf{y}|} \tag{9}$$

for M points $\mathbf{x}_\ell$ and N points $\mathbf{y}_k$. The corresponding algorithm is linear in both the number of evaluation points $\mathbf{x}$ and the number of triangles describing the boundary mesh. If both numbers grow linearly, the effort will grow quadratically.

2.1.2 The fast multipole method

The fast multipole method (FMM) [8, 9, 28] will be used to reduce this quadratic effort for the evaluation of the functional $J(\mathbf{x})$ by introducing a systematic approximation. The fast multipole method is used frequently in boundary element methods, e.g. in [21, 22, 23]. For an overview of the FMM see e.g. [19].

If a separation of the variables $\mathbf{x}$ and $\mathbf{y}$ is possible for the kernel $k(\mathbf{x}, \mathbf{y})$ as in

$$k(\mathbf{x}, \mathbf{y}) = \sum_{i=0}^{\infty} f_i(\mathbf{x}) g_i(\mathbf{y}), \tag{10}$$

the evaluation of $J(\mathbf{x})$ will become cheaper by using an approximation defined by a finite maximum expansion degree p.

The main ingredients of the fast multipole method are the approximation of the kernel function by an appropriate series expansion and the use of a hierarchical structure to compute these expansions efficiently. The series expansion is used to separate the integration variables $\mathbf{x}$ and $\mathbf{y}$ of the kernel function from each other. A first choice would be to use a Taylor series expansion of $k(\mathbf{x}, \mathbf{y})$, as used in the panel clustering method [11], but an expansion based on spherical harmonics is often more suitable than a Taylor series expansion.

Starting from

$$|\mathbf{x} - \mathbf{y}|^2 = (\mathbf{x} - \mathbf{y}) \cdot (\mathbf{x} - \mathbf{y}) = |\mathbf{x}|^2 + |\mathbf{y}|^2 - 2\,|\mathbf{x}|\,|\mathbf{y}|\,(\widehat{\mathbf{x}} \cdot \widehat{\mathbf{y}})$$

with $\widehat{\mathbf{x}} = \mathbf{x}/\,|\mathbf{x}|$, the kernel can be written as

$$k(\mathbf{x}, \mathbf{y}) = \frac{1}{|\mathbf{y}|\sqrt{1 - 2(\widehat{\mathbf{x}} \cdot \widehat{\mathbf{y}})\dfrac{|\mathbf{x}|}{|\mathbf{y}|} + \left(\dfrac{|\mathbf{x}|}{|\mathbf{y}|}\right)^2}}.$$

From the Taylor series expansion of

$$\frac{1}{\sqrt{1-t}} = \sum_{\ell=0}^{\infty} \frac{f^{(\ell)}(0)}{\ell!}(t-0)^\ell = \sum_{\ell=0}^{\infty} \frac{(2\ell)!}{(2^\ell \ell!)^2} t^\ell \qquad \text{for } |t| < 1,$$

where $t = 2(\widehat{\mathbf{x}} \cdot \widehat{\mathbf{y}})\,|\mathbf{x}|\,/\,|\mathbf{y}| - (|\mathbf{x}|\,/\,|\mathbf{y}|)^2$, the expansion of the kernel $k(\mathbf{x}, \mathbf{y})$ in Legendre polynomials

$$k(\mathbf{x}, \mathbf{y}) = \sum_{n=0}^{\infty} P_n(\widehat{\mathbf{x}} \cdot \widehat{\mathbf{y}}) \frac{|\mathbf{x}|^n}{|\mathbf{y}|^{n+1}}$$

can be derived. The Legendre polynomials $P_n(u)$ can be given by

$$P_n(w) = \frac{1}{2^n n!} \frac{\mathrm{d}^n}{\mathrm{d}w^n} \left[(w^2 - 1)^n\right] \qquad \text{for } |w| \leq 1,$$

where $\mathrm{d}^n/\mathrm{d}w^n[\bullet]$ is the n^{th} derivative of the expression in brackets. The accuracy of the approximation using the finite expansion degree p

$$k_p(\mathbf{x}, \mathbf{y}) = \sum_{n=0}^{p} P_n(\widehat{\mathbf{x}} \cdot \widehat{\mathbf{y}}) \left(\frac{|\mathbf{y}|}{|\mathbf{x}|}\right)^n \tag{11}$$

and later of the fast multipole algorithm can be controlled, as the estimate

$$|k(\mathbf{x}, \mathbf{y}) - k_p(\mathbf{x}, \mathbf{y})| \leq \frac{1}{|\mathbf{y}| - |\mathbf{x}|} \left(\frac{|\mathbf{x}|}{|\mathbf{y}|}\right)^{p+1} \qquad \text{for } |\mathbf{y}| > |\mathbf{x}| \tag{12}$$

shows, which can be derived by the use of the geometric series.

The separation of the variables $\mathbf{x}$ and $\mathbf{y}$ is now done by an addition theorem for Legendre polynomials using spherical harmonics [12]. Note that all solutions of the Laplace equation can be given by an expansion in spherical harmonics.

The kernel function will be approximated in terms of an appropriate reformulation, see [24, 36, 38], of the spherical harmonics by the splitting

$$k(\mathbf{x},\mathbf{y}) = \frac{1}{|\mathbf{x}-\mathbf{y}|} \;\approx\; k_p(\mathbf{x},\mathbf{y}) = \sum_{n=0}^{p}\sum_{m=-n}^{n} \overline{S_n^m}(\mathbf{y}) R_n^m(\mathbf{x}) \tag{13}$$

for $\mathbf{x} \neq \mathbf{y}$. It contains the two complex valued terms

$$R_n^{\pm m}(\mathbf{x}) \;=\; \frac{1}{(n+m)!}\frac{\mathrm{d}^m}{\mathrm{d}w^m} P_n(w)\big|_{w=\widehat{x}_3} (\widehat{x}_1 \pm i\widehat{x}_2)^m \,|\mathbf{x}|^n\,, \tag{14}$$

$$S_n^{\pm m}(\mathbf{y}) \;=\; (n-m)!\frac{\mathrm{d}^m}{\mathrm{d}w^m} P_n(w)\big|_{w=\widehat{y}_3} (\widehat{y}_1 \pm i\widehat{y}_2)^m \frac{1}{|\mathbf{y}|^{n+1}} \tag{15}$$

in Cartesian coordinates x_i for $m \geq 0$ and $\widehat{x}_i = x_i/|\mathbf{x}|$. The term $\overline{S}$ in Eq. (13) is the conjugated form of S in Eq. (15). The kernel expansion will be applied to the potential in Eq. (6), where it is integrated over all boundary elements. In the following, we want to control the convergence and the accuracy of the truncated expansion k_p.

Now, the second basic idea of this approach, the building of the hierarchical structure becomes important. In order to do the calculation of the kernel efficiently, the whole system is divided into boxes. Building of the hierarchy can be done from the top down. All boundary elements $\{\tau_k\}_{k=1}^N$ are included in a box containing the original domain of the body Y. The cluster ω_1^0 of level 0 consists of all boundary elements $\{\tau_k\}_{k=1}^N$. The hierarchical structure is build recursively by the refinement of the box corresponding to a cluster of the level λ into eight similar boxes which form the clusters $\omega_j^{\lambda+1}$ of a finer level $\lambda+1$. The clusters $\omega_j^{\lambda+1}$ are called the sons of the father cluster ω_i^λ and consist of all the boundary elements τ_k whose center is inside the corresponding box. This refinement is done until a minimum number of triangles is reached in the cluster or until a maximum level L is reached. Each of the boundary elements τ_k is assigned to the cluster ω_i^L which contains the mid point of τ_k. In this paper, we restrict ourselves to the case of a regular distribution of the boundary elements $\{\tau_k\}_{k=1}^N$ of a globally quasi uniform mesh. Empty clusters containing no boundary elements are neglected. Nevertheless, the method can be extended to the adaptive case, see for example [4, 17].

Since we apply this method to collision detection, we do not know the position of the points of X beforehand. Also, the box σ_1^0 containing these evaluation points $\mathbf{x}$ may differ from the box ω_1^0 containing the original domain Y. Therefore, a second independent cluster tree is build. If the evaluation points were known at setup time, their distribution could be used to build a suitable cluster tree. A second possibility, which has to be used here, is to construct a cluster tree without any information about the number of evaluation points and their distribution. Only the extensions of the evaluation domain σ_1^0 are needed. Then the clusters σ_j^λ are constructed recursively in a similar way as described before.

In the following, we will call clusters ω_i^λ to be in the "nearfield" of a cluster σ_j^λ, if they are located close to the box corresponding to σ_j^λ. Clusters are called to be in the "farfield", whose distance is larger.

Since we are interested in the relation of a point $\mathbf{x}$ to triangles $\tau_k \subset \Gamma = \partial Y$ that are inside different boxes of different cluster trees, the following criterion is used in order to identify whether a triangle is in the nearfield or in the farfield of a point. A cluster ω_i^λ on the same level λ is called to be in the nearfield of the cluster σ_j^λ if the condition

$$\frac{\operatorname{dist}\left\{C(\omega_i^\lambda), C(\sigma_j^\lambda)\right\}}{\max\left\{r(\omega_i^\lambda), r(\sigma_j^\lambda)\right\}} \leq (d+1) \tag{16}$$

is satisfied with a suitably chosen parameter $d > 1$. Here, $C(\bullet)$ denotes the center of the box which contains the corresponding cluster, and $r(\bullet)$ is the corresponding cluster radius, i.e. $r(\omega_i^\lambda) = \sup_{\mathbf{x}\in\omega_i^\lambda} \left|\mathbf{x} - C(\omega_i^\lambda)\right|$. Moreover, the nearfield of a father cluster $\sigma_i^{\lambda-1}$ must contain the nearfields of all its own sons $\sigma_j^\lambda \subset \sigma_i^{\lambda-1}$.

Note that the series expansion in Eq. (13) is not valid in the nearfield due to Eq. (12). Therefore, the approximation of the functional $J(\mathbf{x}_\ell)$ for $\ell = 1, \dots, M$ now reads

$$\widetilde{J}(\mathbf{x}_\ell) = -\sum_{k\in\mathrm{NF}(\mathbf{x}_\ell)} J_k(\mathbf{x}_\ell) - \sum_{k\in\mathrm{FF}(\mathbf{x}_\ell)} \frac{1}{4\pi}\int_{\tau_k} \mathbf{n}_\mathbf{y}\cdot\nabla_\mathbf{y} \sum_{n=0}^{p}\sum_{m=-n}^{n} \overline{S_n^m}(\mathbf{y}) R_n^m(\mathbf{x}_\ell)\mathrm{d}s_\mathbf{y}, \tag{17}$$

where

$$J_k(\mathbf{x}) := \frac{1}{4\pi}\int_{\tau_k} \mathbf{n}_\mathbf{x}\cdot\nabla_\mathbf{x}\frac{1}{|\mathbf{x}-\mathbf{y}|}\mathrm{d}s_\mathbf{x} \tag{18}$$

is computed exactly. The set of indices of triangles in the nearfield of the cluster σ_j^λ containing $\mathbf{x}_\ell$ due to the admissibility condition (16) is $\mathrm{NF}(\mathbf{x}_\ell)$, and $\mathrm{FF}(\mathbf{x}_\ell)$ is its counterpart. Next, the expression

$$L_n^m(O,k) := \int_{\tau_k} \mathbf{n}_\mathbf{y}\cdot\nabla_\mathbf{y}\overline{S_n^m}(\mathbf{y})\mathrm{d}s_\mathbf{y} \tag{19}$$

is used, which contains all coefficients related to the variable $\mathbf{y}$. Expression (19) is defined with respect to the origin O of some local coordinate system. Then, utilizing the fact that the integration only has to be done over the parts of the integration term containing $\mathbf{y}$, the approximation of the functional can be written in the form

$$\begin{aligned}\widetilde{J}(\mathbf{x}_\ell) &= -\sum_{k\in\mathrm{NF}(\mathbf{x}_\ell)} J_k(\mathbf{x}_\ell) - \frac{1}{4\pi}\sum_{n=0}^{p}\sum_{m=-n}^{n} R_n^m(\mathbf{x}_\ell)\sum_{k\in\mathrm{FF}(\mathbf{x}_\ell)} L_n^m(O,k)\\ &= -\sum_{k\in\mathrm{NF}(\mathbf{x}_\ell)} J_k(\mathbf{x}_\ell) - \frac{1}{4\pi}\sum_{n=0}^{p}\sum_{m=-n}^{n} R_n^m(\mathbf{x}_\ell)\widetilde{L}_n^m(O,\mathrm{FF}(\mathbf{x}_\ell)),\end{aligned} \tag{20}$$

with the sum of all expansions of the farfield

$$\widetilde{L}_n^m(O,\mathrm{FF}(\mathbf{x}_\ell)) = \sum_{k\in\mathrm{FF}(\mathbf{x}_\ell)} L_n^m(O,k) \quad \text{for } n = 0,\dots,p, \quad m = -n,\dots,n. \tag{21}$$

The coefficients $L_n^m(O,k)$ can either be computed exactly [17, 18] or can be approximated by the use of some numerical quadrature rule.

For each $\mathbf{x}_\ell$, $\ell = 1,\ldots,M$, the evaluation of the approximate functional $\widetilde{J}(\mathbf{x}_\ell)$ requires the exact evaluation of the small nearfield part. If the coefficients $\widetilde{L}_n^m(O,\mathrm{FF}(\mathbf{x}_\ell))$ of the farfield part are known, the evaluation of the functional $\widetilde{J}(\mathbf{x}_\ell)$ will be very fast. The global coefficients $\widetilde{L}_n^m(O,\mathrm{FF}(\mathbf{x}_\ell))$ depending on ℓ can be efficiently evaluated from the local coefficients $L_n^m(O,k)$ by using the artificial hierarchies described before.

The efficient computation of the coefficients $\widetilde{L}_n^m(O,\mathrm{FF}(\mathbf{x}_\ell))$ in Eq. (21) will be described only very briefly. More detailed descriptions can be found, e.g., in [8, 9].

The following steps only have to be done once as a preprocessing step, independent on the number of evaluation points. In a system containing several moving bodies, such a hierarchy is built for each body and stored in its body fixed coordinate system.

The farfield of each cluster σ_j^L will be described by clusters ω_i^λ which are as large as possible and which can be used for many different farfields. For each combination of the clusters σ_j^L and the corresponding farfield clusters ω_i^λ the local coefficients are now calculated to describe the effect of the farfield. In order to do this efficiently, as much information as possible is shared when these coefficients are determined. Therefore, as a first step, the coefficients of the multipole expansions

$$\widetilde{M}_n^m(C(\omega_j^L),P(\omega_j^L)) = \sum_{\tau_k\in\omega_j^L} M_n^m(C(\omega_j^L),k) = \sum_{\tau_k\in\omega_j^L} \int_{\tau_k} \mathbf{n}_\mathbf{y}\cdot\nabla_\mathbf{y} R_n^m(\mathbf{y})\mathrm{d}s_\mathbf{y} \qquad (22)$$

are computed for all clusters ω_j^L on the finest level L, where $\mathrm{P}(\omega_j^\lambda) := \{k, \tau_k \in \omega_j^\lambda\}$ denotes the index set of the panels τ_k belonging to the cluster ω_j^λ, $C(\omega_j^\lambda)$ denotes the center of the cluster ω_j^λ and of the local coordinate system, described as O before. Next, these coefficients are translated to their common fathers up to the coarsest level by

$$\begin{aligned}\widetilde{M}_n^m(C(\omega_j^\lambda),\mathrm{P}(\omega_j^\lambda)) \;=\; & \sum_{\omega_i^{\lambda+1}\in\mathrm{Sons}(\omega_j^\lambda)} \sum_{s=0}^{n}\sum_{t=-s}^{s} R_s^t(\overrightarrow{C(\omega_j^\lambda)C(\omega_i^{\lambda+1})}) \\ & \qquad \widetilde{M}_{n-s}^{m-t}(C(\omega_i^{\lambda+1}),\mathrm{P}(\omega_i^{\lambda+1})).\end{aligned} \qquad (23)$$

These coefficients, related to a cluster ω_j^λ, contain all the contributions of the triangles contained in ω_j^λ to the functional $\widetilde{J}(\bullet)$. They can be derived from Eq. (13) by interchanging $\mathbf{y}$ and $\mathbf{x}_\ell$.

Next, these multipole expansions are converted into local expansions of σ_j^λ by

$$\widetilde{L}_n^m(C(\sigma_j^\lambda),\mathrm{P}(\omega_j^\lambda)) = \sum_{s=0}^{\infty}\sum_{t=-s}^{s}(-1)^n\overline{S_{n+s}^{m+t}}(\overrightarrow{C(\omega_i^\lambda)C(\sigma_j^\lambda)})\widetilde{M}_s^t(C(\omega_i^\lambda),\mathrm{P}(\omega_j^\lambda)). \qquad (24)$$

These conversions are executed on the highest level possible, i.e., the level where two clusters σ_j^λ and ω_i^λ are in the farfield of each other but their fathers $\sigma_j^{\lambda-1}$ and $\omega_i^{\lambda-1}$ are still in their mutual nearfield.

In the last step of the preprocessing, the local expansions are translated from the clusters σ_i^λ to their sons $\sigma_j^{\lambda+1}$ by

$$\widetilde{L}_n^m(C(\sigma_j^{\lambda+1}), \mathrm{FF}(\omega_i^\lambda)) = \sum_{s=n}^{p} \sum_{t=-s}^{s} R_{s-n}^{t-m}(\overrightarrow{C(\sigma_i^\lambda)C(\sigma_j^{\lambda+1})}) \widetilde{L}_s^t(C(\sigma_i^\lambda), \mathrm{FF}(\omega_i^\lambda)) \tag{25}$$

and summed until the finest level L is reached. The derivation of the translation and conversion formulae in terms of addition theorems for the reformulated spherical harmonics can be found, e.g., in [36, 38].

Now, the coefficients $\widetilde{L}_n^m(\mathrm{FF}(O, \mathbf{x}_\ell))$ are provided for an efficient evaluation of the approximate functional in Eq. (20). In the evaluation phase, only the expansion and the nearfield part have to be evaluated.

2.1.3 Complexity of the algorithm

The algorithm based on the functional J without using the fast multipole method is of order $\mathcal{O}(NM)$. Using the fast multipole method introduces an error in the evaluation of the functional $\widetilde{J}$. This error can be controlled by the expansion degree p. The error introduced by an eventually used numerical integration formula can easily be estimated and is neglected here. We also restrict the analysis to the case of quasi–uniform boundary element meshes.

For simplicity, we assume that the clusters ω_i^λ and σ_j^λ on the level λ are of about the same size. If the sizes of the domain Y and of the evaluation domain X differ significantly, extra levels can be introduced in the top of the corresponding cluster tree to guarantee this feature. In what follows, we assume a global cluster radius r for all clusters on the finest level. The error estimate

$$\left| \frac{\partial}{\partial \mathbf{n_y}} (k(\mathbf{x}, \mathbf{y}) - k_p(\mathbf{x}, \mathbf{y})) \right| \;\leq\; \frac{1+\pi}{(d-1)r^2} \left(p + 1 + \frac{d}{d-1} \right) \left(\frac{1}{d} \right)^{p+2} \tag{26}$$

see [22], for $|\mathbf{x}| \leq r$ and $|\mathbf{y}| \geq d\, r$, see Eq. (16), gives the possibility to estimate the error introduced by the fast multipole method as

$$\begin{aligned} \left| J(\mathbf{x}) - \widetilde{J}(\mathbf{x}) \right| \;&=\; \left| \sum_{k=1}^{N} \int_{\tau_k} \mathbf{n_y} \cdot \nabla_{\mathbf{y}} \left(k(\mathbf{x}, \mathbf{y}) - \widetilde{k}(\mathbf{x}, \mathbf{y}) \right) \mathrm{d}s_{\mathbf{y}} \right| \\ &\leq\; \Delta_\Gamma \frac{1+\pi}{(d-1)r^2} \left(p + 1 + \frac{d}{d-1} \right) \left(\frac{1}{d} \right)^{p+2} . \end{aligned} \tag{27}$$

While the described approach only gives an approximation $\widetilde{J}(\mathbf{x})$ of the functional $\widetilde{J}(\mathbf{x})$, one needs to control the corresponding error. It is suitable to choose $p \sim \log N$ as the cluster radius r is proportional to h, while $h = \sqrt{\Delta_\tau}$ denotes the mesh size of the boundary elements.

In our complexity analysis, the parameter d is fixed and the expansion degree $p \sim \log N$ ensures the optimal error estimate. The number of boundary elements belonging to a cluster ω_i^L on the finest level is chosen to be of order $\mathcal{O}(\log^2 N)$. So the number of clusters on the finest level of the mesh cluster tree is $N_{\omega,L} \in \mathcal{O}(N/\log^2 N)$. Also the total number of clusters of the mesh cluster tree is $N_\omega \in \mathcal{O}(N/\log^2 N)$ as an octree is used.

For the estimate of the number of clusters in the evaluation tree, a simplification is used in the construction of the tree. In contrast to the method described before, the clusters σ_j^λ are refined until the clusters do not have any clusters ω_i^λ in their nearfield or the maximum cluster level L is reached. This simplification produces a larger number of clusters σ_j^λ, so the procedure described before will only perform better in setting up the expansions in the fast multipole method. Both cases will be included in the estimate of the effort of the evaluation part. For any cluster ω_i^λ, a fixed maximal number of clusters σ_j^λ of the same level λ is constructed. This number is given by the sons of all nearfield clusters $\sigma_j^{\lambda-1}$ of the father cluster $\omega_i^{\lambda-1}$ and estimated by c_d. The total number of clusters in the evaluation tree can be estimated by $N_\sigma \leq c_d N_\omega$. This is highly overestimated, as most of the clusters σ_j^λ are counted many times, but this estimate is sufficient for an asymptotic complexity estimate. The translations and conversions of the expansions are of order $\mathcal{O}(p^4)$ as the expansions have order $\mathcal{O}(p^2)$ coefficients. In Table 1 the efforts for the different operations are collected.

The total effort is then given by $\mathcal{O}(N \log^2 N) + \mathcal{O}(M \log^2 N)$ and, therefore, is almost linear in the number of boundary elements and in the number of evaluation points. When using a variable version [29, 33] of the fast multipole method the logarithmic terms could be dropped in the asymptotic complexity estimate.

2.2 Ray crossing method

A second approach that can be used in order to check whether nodes lie within a polyhedron is the ray crossing method, see [20]. This approach is completely different and has its origin in traditional computational geometry and computer graphics. There, a ray to infinity originating from the observed point is created, see Figure 1. Then the intersections of this ray with the surface of the body are counted.

Point P is inside the polyhedron, if the number of intersections of an arbitrary ray with the surface of the polyhedron is odd, as shown in Figure 1. Also it can be said, that a point, e.g. point Q in Figure 1, is outside of the polyhedron, if the number of intersections is even for arbitrary rays from this point (where 0 is considered to belong to the even numbers). In order to perform that check, the ray has to be tested against all separate surface parts (faces f) of the body. The problem about counting these ray crossings is the wide variety of possible degeneracies that could occur, compare [20]. These degeneracies could be, e.g., a ray laying in a face, hitting a vertex or an edge, lying collinerarly to an edge, etc. However, the treatment of such degeneracies must not be taken into account for the implementation. Since the direction of the ray is completely arbitrary and the creation of another ray is not time consuming, it is simply checked for each ray, whether

Table 1: Complexity of the fast multipole approach

operation and explanation	costs
simple approach	
computation straight forward	$\mathcal{O}(NM)$
setting up all expansions	
computing the coefficients up to degree p for all N boundary elements	$N\mathcal{O}(p^2)$
translation of the multipole expansion of all N_ω clusters to their fathers	$N_\omega\mathcal{O}(p^4)$
For all N_ω clusters the conversions of multipole expansions into local expansions of the clusters of the evaluation tree have to be applied. The number of conversions per cluster is again bounded by c_d.	$c_d N_\omega\mathcal{O}(p^4)$
translation of the local expansion to all N_σ clusters from their fathers	$N_\sigma\mathcal{O}(p^4)$
evaluation of the functional $\widetilde{J}$	
evaluation of the local expansions of degree p for all M evaluation points	$M\mathcal{O}(p^2)$
the remaining nearfield part of one evaluation point consists of at most $\mathcal{O}(\log^2 N)$ boundary elements	$M\mathcal{O}(\log^2 N)$

such degeneracies exist, and if so, a new ray is created with random direction. Such a ray to infinity can be substituted by a line segment PR that is long enough, if its endpoint R is definitely outside a bounding box around the body.

The collision test then is accomplished in several steps. In a first step, all points of the tested body that are not positioned within the bounding box of the other polyhedron are discarded. For the planar example shown in Figure 2 only six nodes of body 2 are left to be investigated instead of the original twelve nodes.

For these remaining nodes, the random rays have to be generated and tested against all faces of body 1. Such a triangular face f consists of three nodes, F_1, F_2, and F_3, see

Figure 3. The example in this figure has 20 such triangular faces (ten of which are shown in the figure, and ten more are back facing). For this part of the test now bounding boxes

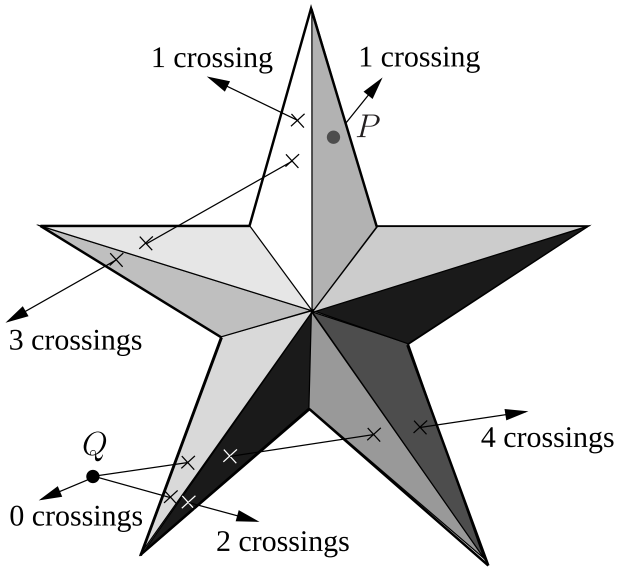

Figure 1: Collision detection by means of checking a ray for each observed point of another body. Here, three possible rays are shown for a point P that is inside the body (one or three crossings), and for a point Q that is located outside the body (no, two, and four crossings).

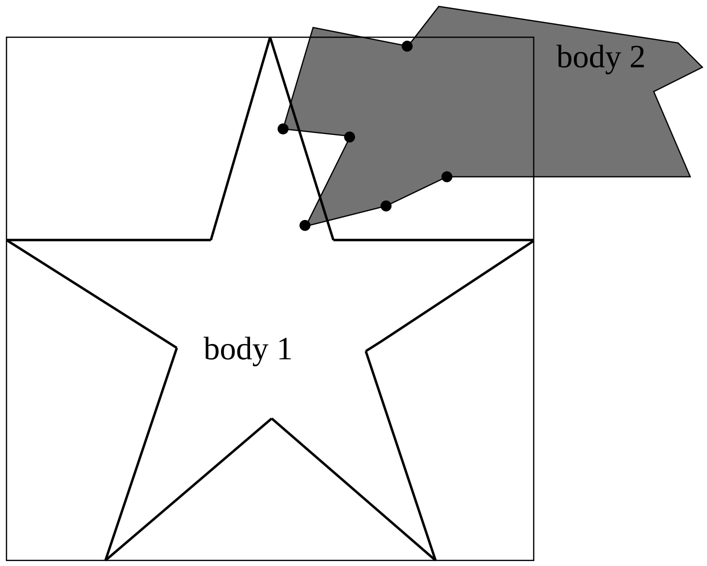

Figure 2: Only points that are positioned within the bounding box are considered.

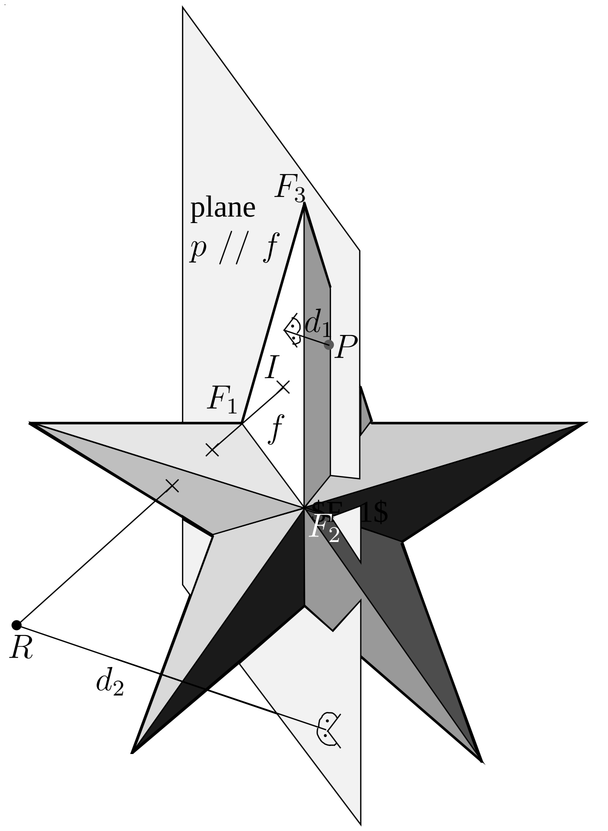

Figure 3: Non-convex polyhedron with face f (F_1, F_2, F_3), and a point P of another body lying within the polyhedron. The distance between P and f is d_1, d_2 is the distance from P to R (in normal direction of f), and I is the intersection point of line PR with f.

are laid around each of these faces. Then the ray is first tested whether it crosses the bounding box around f. For each face, whose bounding box is crossed, a further test has to be accomplished. It is clear that a ray cannot cross f if they are parallel. For each case, where the ray crosses the bounding box of f and where the ray is not parallel to f, the ray definitely crosses the plane, in which f is located. By controlling, whether such an intersection point of the ray with the plane of a surface really lies on the surface of the body, i.e. on the triangle f, the ratio of d_1 (which is the distance between the observed point to the plane) to d_2 (that is the distance in normal direction between R and P) has to be considered, see Figure 3. This ratio can be determined as

$$r = \frac{(\mathbf{F} - \mathbf{P}) \cdot \mathbf{n}}{(\mathbf{R} - \mathbf{P}) \cdot \mathbf{n}} = \frac{d_1}{d_2}, \tag{28}$$

see [20, 35]. Here, $\mathbf{P}$ and $\mathbf{R}$ are the position vectors to the points P and R, respectively, and $\mathbf{F}$ is the position vector to a point of face f. This point is therefore in the plane of f. If d_1 equals zero, it is clear that the observed point, too, is located on the plane of f, and it has to be checked, whether the point is located within the boundary of triangle f. If d_2 equals zero and d_1 is not equal to zero, then this ray is parallel to f and, therefore, never crosses it. In this case, the ray can be tested against the other faces. The case that d_1 and d_2 are zero at the same time is one of the earlier mentioned degeneracies – in this case a new ray is generated and tested against all faces again. If neither d_1 nor d_2 are equal to zero, the ratio r is computed.

By means of this ratio, the position vector of I, which is the intersection point of PR with the plane, can be computed,

$$\mathbf{I} = \mathbf{P} + r\,(\mathbf{R} - \mathbf{P}). \tag{29}$$

The meaning of Eq. (29) is clear if one adopts the theorem on intersecting lines, see Figure 3. That means, r of Eq. (28) is the ratio of d_1 to d_2 that is equal to the ratio of PI to PR.

If r is negative or greater than 1, then point I is not on f and the next face can be tested. For $r = 1$, I would be equal to R, but since R is outside the body, in this case, too, I is not located on f. That means, only in cases where $r = 0$ (i.e. $d_1 = 0$) and $0 < r < 1$, it is possible that I (or P) are lying on f.

To check whether P is located on f, both, P and f are projected in normal direction of f onto a plane, see Figure 4.

Then the position of the projected point P' to the projected face is determined by checking its position versus the edges of the projected triangle. This is shown on the right side of Figure 4. For each area the "Area-Signs" are calculated. That means, it is checked whether a point P' that is possibly within that area is to the left or to the right of the three edges that form the triangle. If P' is to the right of any of these edges (that are directed counter clockwise around f), the projection is not inside f, and, therefore, the ray does not cross f. In this case, the next face can be tested. Only if the point is to the left of all three edges, P' is located on f. Also cases, where P is lying on a vertex or edge of f,

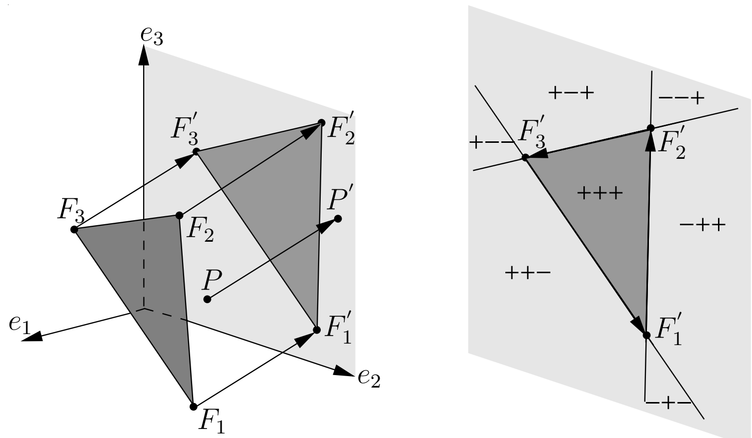

Figure 4: Projection of a face (left), signs received for the projected point in the special areas (right), see [20, 35].

can be determined. The results for the "Area-Signs" along the projections would then be $0--$, $-0-$, or $--0$ for the edges and $0-0$, $00-$, and -00 for the vertices. Then, if one of these six cases or $+++$ is valid, the other faces do not have to be tested anymore, since it is directly clear, that the point is located on f and, hence, on the surface of the body. However, if none of these seven cases is valid, P is not located on f or its boundaries, and the other faces of the body have to be tested.

Still, we have to handle the case that the ratio of Eq. (28) is $0 < r < 1$. In this case, similar to the case $r = 0$, it has to be tested, whether I (the intersection point of the ray with the plane of f) is located on f or outside the boundaries of the triangle. In order to determine this, a "Volume-Sign" is calculated, that is very similar to the "Area-Sign" explained above. Here, the location of the segment PR to the face f is of interest. The signed volumes of the three tetrahedra defined by PR and each edge of f are determined. Again it is detected, whether "Volume-Signs" are negative for any of these tetrahedra. Figure 5 shows three exemplary cases, where the "Volume-Sign" of the three volumes are shown.

The "Volume-Signs" for the cases shown in Figure 5 are $+++$, $+0+$ and $+00$. Again, if one or more of the determined "Volume-Signs" are negative, then I is outside of f. That means that the segment PR does not cross f, and the next faces of the body can be determined. If one or more of the "Volume-Signs" are zero, as for the two right examples of Figure 5, then it is not directly clear to which face the ray belongs. Such a case is considered as a degeneracy, and a new ray is created, that has to be tested against all faces again. If, however, all three "Volume-Signs" are positive, then it is clear that I is located on f, and, therefore, the ray crosses this face. The total number of such crossings between the ray and the faces is of interest. Therefore, this whole procedure has to be done for all faces.

Finally it can be said that either this determination has stopped in between with the result, that P is located on the surface of the body, or we get a number of crossings of

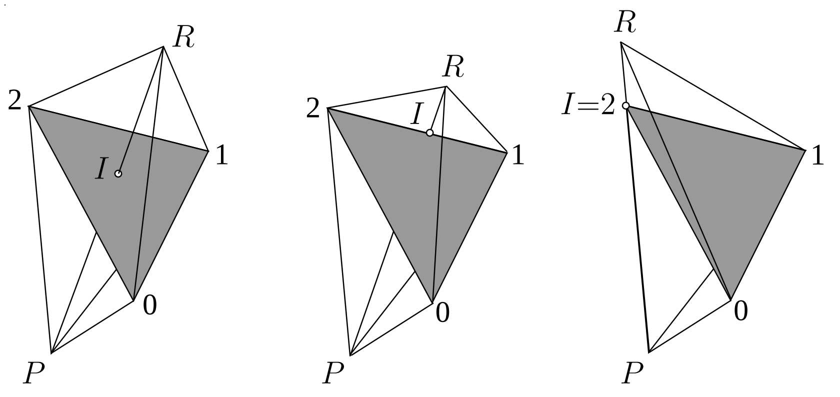

Figure 5: "Volume-Signs" of three cases, where I is inside f, on an edge of f and on a vertex of f, see [20].

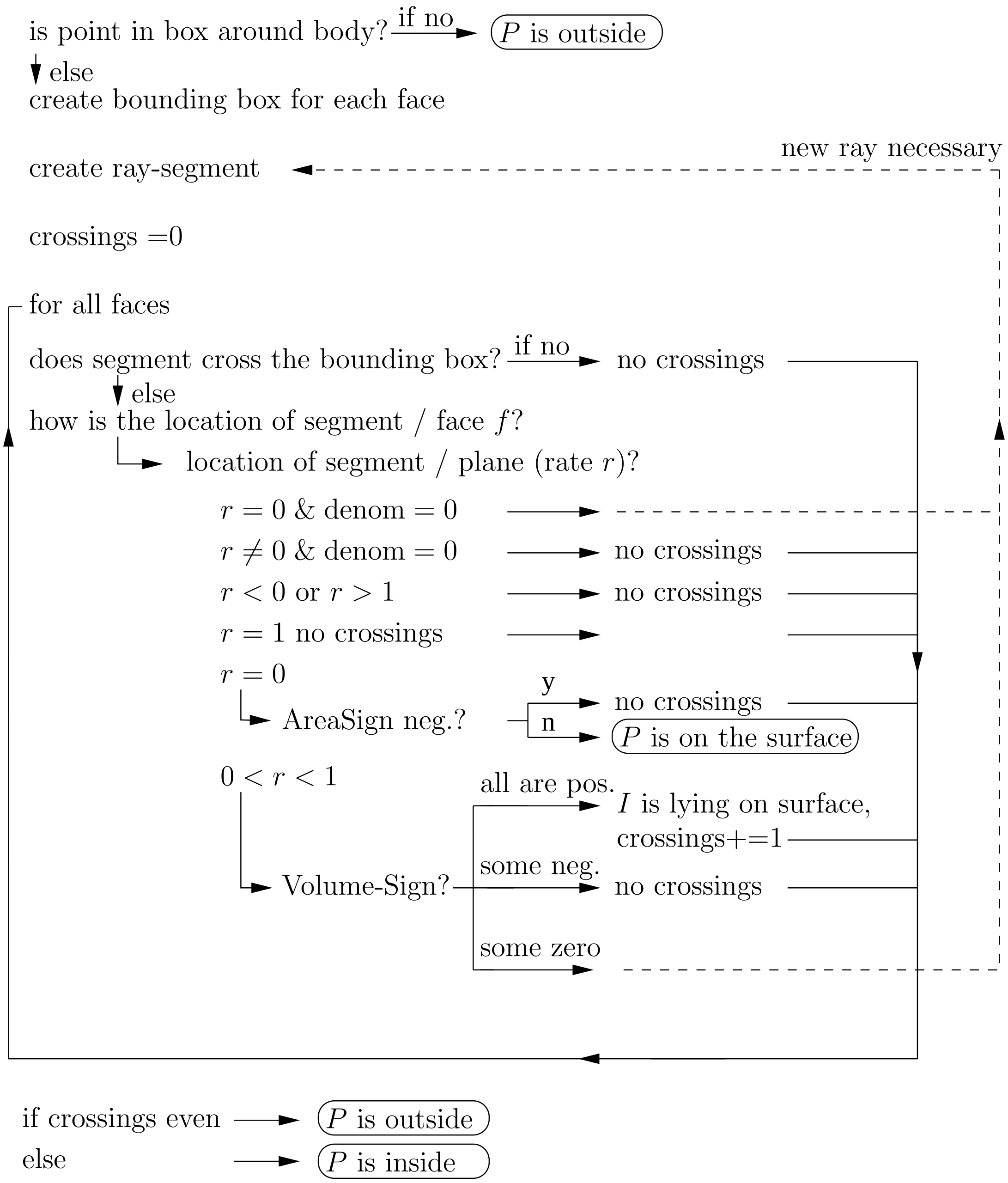

Figure 6: Overview of the ray crossing method.

the segment PR with the surface of the body. If this number is odd, P is lying inside the body, and if it is even or zero, the point is outside the body, as it is shown in Figure 1.

A summary of the whole process is shown in Figure 6. This whole procedure has to be carried out for each point of two neighboring body-pairs that is inside the bounding box of the other body.

3 Comparisons and results

The techniques introduced shall be compared with respect to simulation times for various but always high numbers of faces and testpoints. In order to investigate solely the dependence on the simulation time for different system sizes, several test series are used. The goal of these series is, to keep some system traits as constant as possible and to change only some well defined influencing factors.

The program used for the comparisons is written in C. All program runs were performed on the same 2.8GHz Linux PC that features a working memory of 1 GB. Both methods give correct results and work, therefore, reliably.

3.1 Comparison of the methods using a sphere

In this first, more academic numerical example, approximations of the unit sphere are used for the body Y. The coarsest approximation of the sphere is built by eight triangles. Then these triangles are refined uniformly and the created new nodes are projected onto the surface of the sphere. In this way, seven refinement steps are executed until the finest approximation of the sphere is reached consisting of 131072 triangles. A regular distribution of test points in the surrounding cube is used as scattered grid X. In particular 10^3, 20^3, 40^3, and 80^3 points are used. This setting gives the possibility to compare the two methods and to observe several special features of the fast multipole approach.

First, the results obtained by the fast multipole method are discussed. There, the choice of the maximum level of the cluster tree is important for the performance of the method. With the usually not available knowledge of the number of triangles as well as the number of evaluation points, the depth of the cluster tree can be chosen quite well in advance. In the following this will be called well chosen (WC) cluster tree. The results obtained by means of these choices are shown in Figure 7.

These results are better than the results that can be obtained with a tree depth only depending on the number of triangles. Since in collision detection applications it is usually not known, which and how many points one has to test against the body, in Figure 8 the results which have been obtained with a depth only depending on the number of triangles are shown. In the following this will be called pre-chosen (PC) cluster tree.

At a first sight, these results may seem surprisingly. As the number of evaluation points is increased by a factor of eight with each grid and the number of triangles is increased by a factor of four in each refinement step, one would expect to see these factors in the results due to the asymptotic analysis in Section 2.1.2. However, the results in Figure 7 seem to be better than linear. In the following a linear behavior will be called an asymptotic behavior, so here the results in Figure 7 are not converging asymptotically but behave better than that. The reason is that the total computational time consists of the time for setting up the coefficients of expansion in Eq. (20) and the time for evaluating the expansion and the nearfield part. The time for setting up the coefficients depends only on the number of triangles and on the maximum level of the cluster tree, while the time for the evaluation part depends on the number of evaluation points and on the maximum level of the cluster

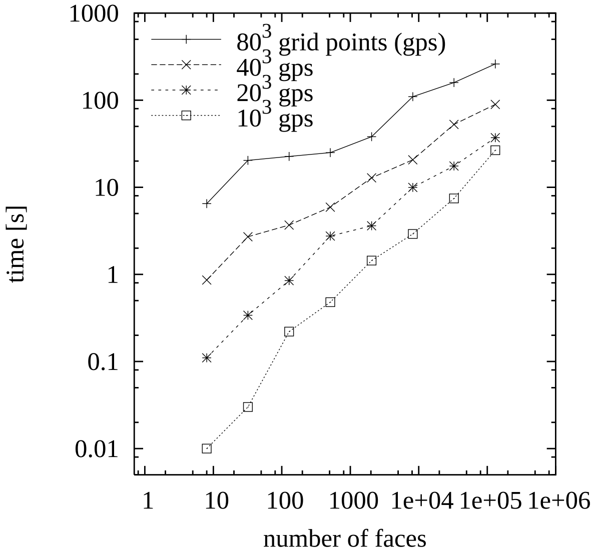

Figure 7: Results of the fast multipole approach using a well chosen (WC) depth of the cluster tree.

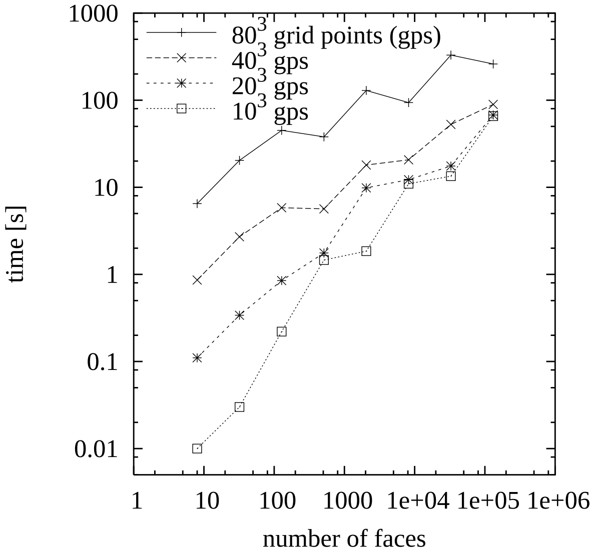

Figure 8: Results of the fast multipole approach using a pre-chosen (PC) depth of the cluster tree.

tree. Therefore, the method is linear in the number of triangles and in the number of evaluation points. As both parts of the computation time depend on the maximum level of the cluster tree, this level is used to balance the two parts and this gives the better results. As long as none of the two numbers is dominating, one does not see the asymptotic, but a better, behavior. The computational times for the grid with only 10^3 evaluation points get closest to the asymptotic linear case as these numbers are dominated by the number of triangles due to the rather low number of evaluation points. If the numbers of triangles and evaluation nodes are both increased by the same factor at the same time, the asymptotic behavior will be seen.

The results in Figure 8 have been obtained using a fixed maximum level of the cluster tree for each of the meshes. When two lines almost touch each other, the total time is dominated by setting up the coefficients of the expansions as the maximum level of the cluster tree is too large for the small numbers of evaluation points. On the other hand, some of the computational times of the grid with 80^3 evaluation points are dominated by the time for evaluating the nearfield part as the depth of the cluster tree is chosen too small. Therefore, the total computational time is reduced sometimes for a refined mesh compared to the previous mesh due to a larger depth of the cluster tree. In the case of the fixed depth of the cluster tree, the linear dependence on the number of evaluation points can be observed in the evaluation part, see Table 2. For example, the shares in computational times for the evaluation part are less than one second, about three seconds, about 24 seconds and 202 seconds in the case of the finest mesh. The numbers also show that the computational times for the mesh of 131072 triangles are reduced in comparison to the times for the mesh of 32768 triangles. This is due to the larger depth of the cluster tree as described before.

Table 2: Computational times in seconds for the evaluation part

# points	8192	32768	131072
10^3	< 1	≈ 1	< 1
20^3	≈ 1	5	3
40^3	10	40	24
80^3	83	329	202

In Figure 9, the computational times obtained by the ray crossing method are shown. The lines show a linear dependency of the method in both the number of triangles and the number of evaluation points. The two different approaches are compared in Figure 10 for the two grids of 10^3 and 80^3 evaluation points.

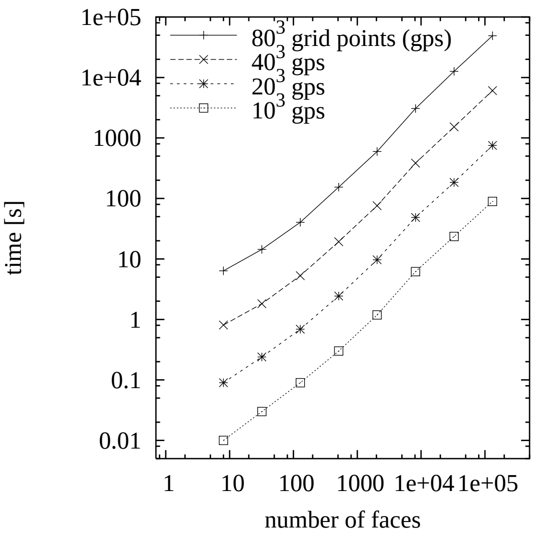

Figure 9: Results obtained by means of the ray crossing method.

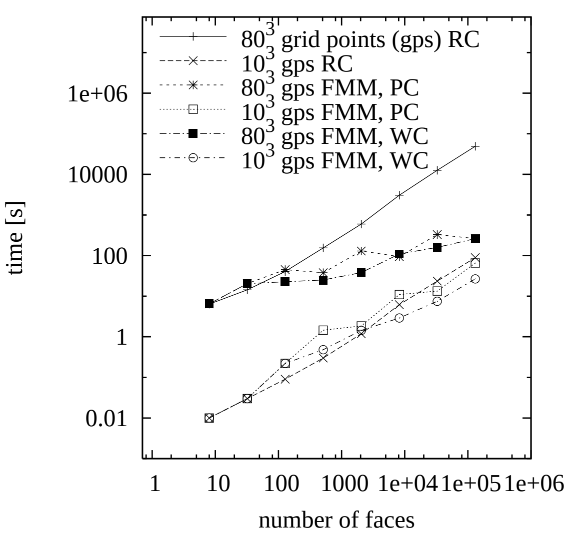

Figure 10: Comparison of both approaches.

In the case of the grid with 10^3 points, the ray crossing method is faster for the meshes with a smaller number of triangles and the fast multipole method is a little bit faster for the finest meshes. In the case of the grid with 80^3 points, the ray crossing method is only faster for the bodies consisting of few triangles. For fine meshes, the fast multipole approach is faster, as the evaluation part is more dominant which is almost independent of the number of triangles in the fast multipole approach. The FMM with a maximum level of the cluster tree depending only on the number of triangles sometimes performs worse than the one with the "optimally" chosen maximum level.

3.2 Comparison using a complicated foam structure

The second example is a foam structure which is approximated by three different meshes consisting of 28952, 120868, and 494124 triangles. These meshes have been provided by Dr. Heiko Andrä, Fraunhofer Institut Techno– und Wirtschaftsmathematik, Kaiserlautern. One of these foams, the one with the smallest mesh, is shown in Figure 11.

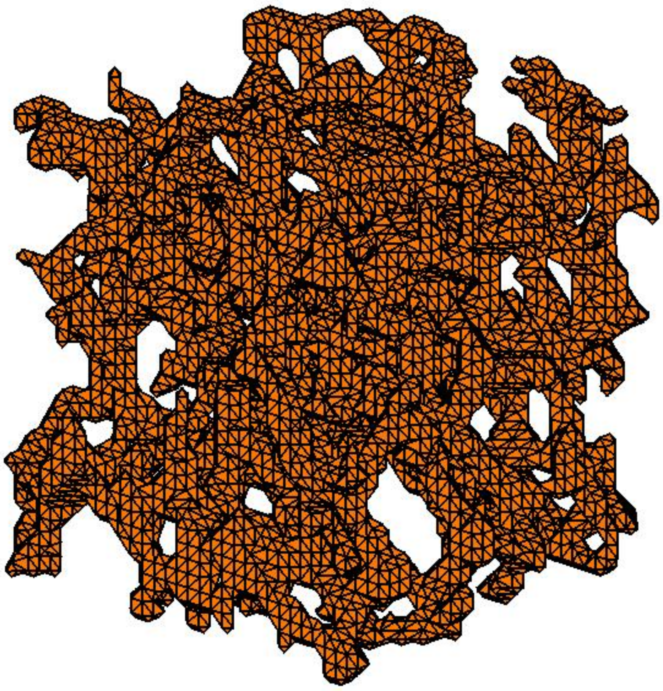

Figure 11: Mesh of the foam.

Now the two collision detection methods are applied again. The same number of test points, 10^3, 20^3, 40^3 and 80^3, are checked whether they are inside or outside these bodies. For the FMM again an optimal choice of cluster tree depth (using knowledge of both, the number of surface triangles and the number of grid points) is used and another tree depth is chosen only by means of the knowledge of the number of triangles. In the case of the FMM with the optimally chosen depth of the cluster tree, a similar performance as for the sphere is obtained, see Figure 12. The computational times increase less than linear with respect to the number of triangles for a fixed number of evaluation points. This is due to the fact that the depth of the cluster tree is used to balance the times for setting up the coefficients and for the evaluation. If the number of triangles and the number of evaluation points are increased at the same time, the asymptotic linear behavior will be observed.

The computational times for the not optimally chosen (PC) maximum level are shown in Figure 13. Again, the results are not as good as for the optimal choice (WC) of the cluster tree depth. Especially the results for the finest mesh are dominated strongly by the

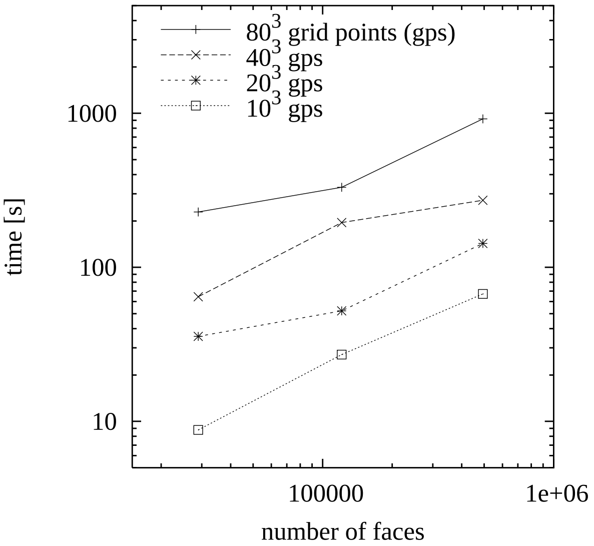

Figure 12: Results of the FMM using a well chosen (WC) cluster tree depth.

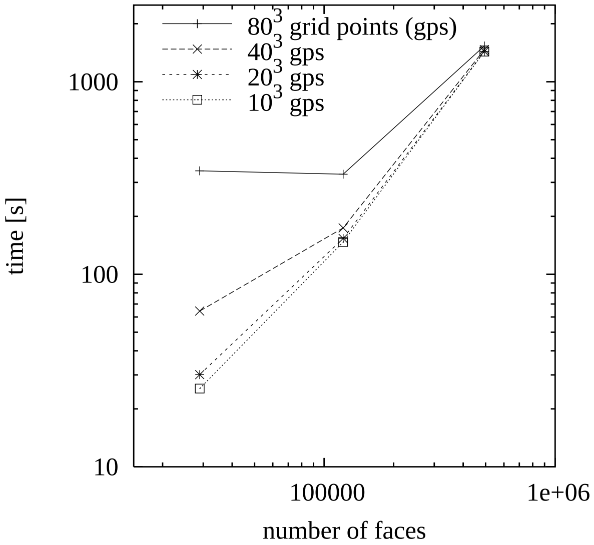

Figure 13: Results of the FMM using a pre-chosen (PC) depth of cluster tree.

time required to set the coefficients since the maximum level is chosen too high due to the huge number of triangles.

For the ray crossing method again the linear dependency of the method can be seen nicely in Figure 14. In the comparison of the different approaches in Figure 15, again the better performance of the fast multipole method for large numbers of faces can be observed. For the smallest grid, the ray crossing method shows a better performance than the results obtained by means of the FMM when there is no knowledge about the numbers of points given. For the large grid, both versions of the FMM show a strongly better performance.

4 Conclusions

In this paper two very different methods that can be used for collision detection have been explained and compared with respect to required computation time. Here, systems containing bodies that are highly complex, i.e. containing many faces, are of interest. In order to investigate the behavior of such systems properly, in the comparisons a scattered grid of points is tested with a body whose complexity is increased.

For the FMM it has to be distinguished between results that are obtained using an optimal choice for the depth of the cluster tree and results where this knowledge is not available. In our example, both, the number of elements of the body as well as the number of points of the scattered grid is known. Having this knowledge, the obtained results are for most cases better than the results obtained by means of the ray crossing method, especially if many faces have to be checked.

However, using this method for a simulation of a system consisting of some bodies, it

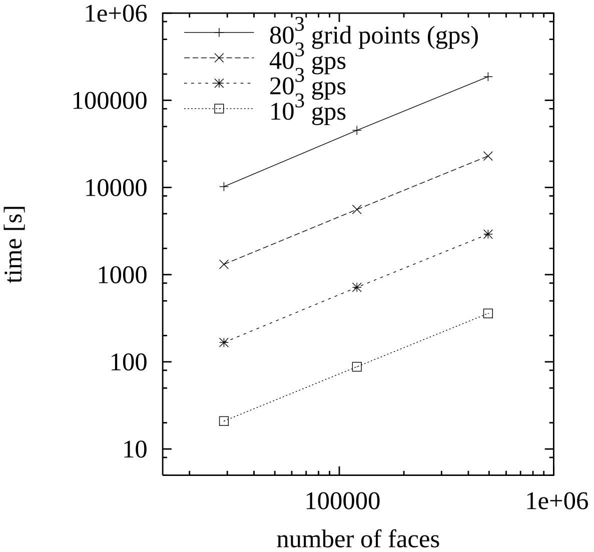

Figure 14: Linear results obtained using the ray crossing method.

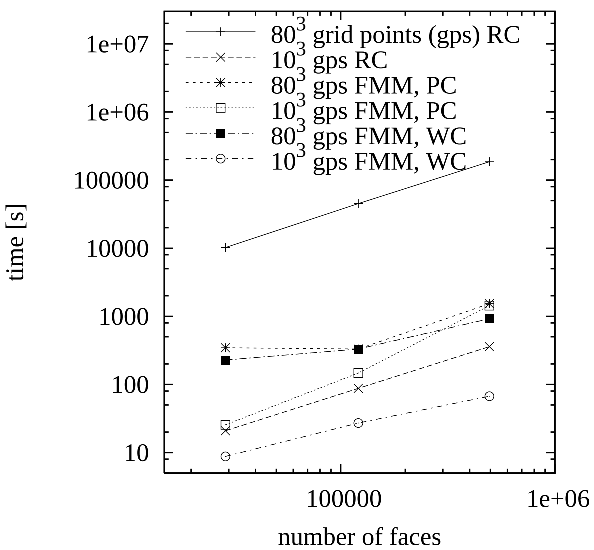

Figure 15: Comparison of both approaches.

is not known from the beginning, how many points have to be tested against a body. The results obtained by using only the knowledge of the complexity of the body are often very similar for all scattered grids, since the chosen tree depth for these cases is only dependent on the number of elements of the body, and is therefore often chosen too high. Comparing the RC method with these results shows that for the foam tested with the small scattered grid, the RC method is more efficient, whereas for the largest scattered grid the FMM method is a lot better than the RC method. For the sphere a similar conclusion can be drawn. Here, for the smallest scattered grid and the five coarsest discretizations of the sphere the RC method is better than the FMM method, and even for the largest scattered grid a better result of the RC method can be obtained for the three smallest spheres. It can be said that both methods are efficient and useful for collision detection purposes. While the FMM has its advantages mainly for very large bodies consisting of a huge number of nodes, the RC method is more appropriate for simpler bodies.

Acknowledgement

The authors have done this research in the collaborative research center SFB 404, working in projects B7 and C10. They want to thank their colleagues within the SFB for the always very nice and helpful atmosphere. This work is financed by the German Research Council DFG, whose support is highly appreciated.

References

[1] M.P. Allen, D.J. Tildesley. Computer Simulations of Liquids. Clarendon Press, Oxford, 1989.

[2] D. Baraff. Dynamic Simulation of Non-Penetrating Rigid Bodies. Ph. D. Thesis, Technical Report 92-1275. Computer Science Department, Cornell University, Ithaka, 1992.

[3] L. Brendel, S. Dippel. Lasting Contacts in Molecular Dynamics Simulations. In: H.J. Herrmann, J.-P. Hovi and S. Luding eds., Physics of Dry Granular Materials - NATO ASI Series E 350 Kluwer Academic Publishers, Dordrecht, (1998) 313-318.

[4] H. Cheng, L. Greengard, V. Rokhlin. A Fast Adaptive Multipole Algorithm in Three Dimensions. Journal of Computational Physics 155 (1999) 468–498.

[5] P.A. Cundall, O.D.L. Strack. A Discrete Numerical Model for Granular Assemblies. Geotechnique 29, 1 (1979) 47–65.

[6] P. Eberhard. Kontaktuntersuchungen durch hybride Mehrkörpersystem / Finite Elemente Simulationen. Shaker, Aachen, 2000.

[7] W.D. Elliot, J.A. Board. Fast Fourier Transform Accelerated Fast Multipole Algorithm. SIAM Journal on Scientific Computing 17 (1996) 398–415.

[8] L. Greengard. The Rapid Evaluation of Potential Fields in Particle Simulation. MIT Press, Cambridge, 1987.

[9] L. Greengard, V. Rokhlin. A Fast Algorithm for Particle Simulations. Journal of Computational Physics 73 (1987) 325–348.

[10] L. Greengard, V. Rokhlin. A New Version of the Fast Multipole Method for the Laplace Equation in Three Dimensions. Acta Numerica 6 (1997) 229–269.

[11] W. Hackbusch, Z.P. Nowak. On the Fast Matrix Multiplication in the Boundary Element Method by Panel Clustering. Numerische Mathematik 54, 4 (1989), 463–491.

[12] E.W. Hobson. The Theory of Spherical and Ellipsoidal Harmonics. Chelsea, New York, 1955.

[13] K.H. Hunt, F.R.E. Grossley. Coefficient of Restitution Interpreted as Damping in Vibroimpact. ASME Journal of Applied Mechanics 7 (1975) 440-445.

[14] Y. Kishino, ed. Powders & Grains. Balkema, Rotterdam, 2001.

[15] S. Luding. Collisions & Contacts between Two Particles. In: H.J. Herrmann, J.-P. Hovi, S. Luding eds. Physics of Dry Granular Media - NATO ASI Series E 350 Kluwer Academic Publishers, Dordrecht, (1998) 285-304.

[16] B. Muth, M.-K. Müller, P. Eberhard, S. Luding. Contacts between Many Bodies. Machine Dynamics Problems 28, 1 (2004) 101–114.

[17] K. Nabors, F.T. Korsmeyer, F.T. Leighton, J. White. Preconditioned, Adaptive, Multipole-Accelerated Iterative Methods for Three-Dimensional First-Kind Integral Equations of Potential Theory. SIAM Journal on Scientific Computing 15 (1994) 713–735.

[18] J.N. Newman. Distributions of Sources and Normal Dipoles over a Quadrilateral Panel. Journal of Engineering Mathematics 20 (1986) 113–126.

[19] N. Nishimura. Fast Multipole Accelerated Boundary Integral Equations Methods. Applied Mechanics Reviews 55 (2002) 299–324.

[20] J. O'Rourke. Computational Geometry in C, 2. Edition. Cambridge University Press, Cambridge, 1998.

[21] G. Of, O. Steinbach. A Fast Multipole Boundary Element Method for a Modified Hypersingular Boundary Integral Equation. In: M. Efendiev, W.L. Wendland eds. Proceedings of the International Conference on Multifield Problems. Lecture Notes in Applied Mechanics, 12. Springer, Berlin, (2003) 163–169.

[22] G. Of, O. Steinbach, W.L. Wendland. The Fast Multipole Method for the Symmetric Boundary Integral Formulation. IMA Journal of Numerical Analysis 26 (2006) 272–296.

[23] G. Of, O. Steinbach, W.L. Wendland. Applications of a Fast Multipole Galerkin Boundary Element Method in Linear Elastostatics. Computing and Visualization in Science 8 (2005) 201–209.

[24] J.M. Perez–Jorda, W. Yang. A Concise Redefinition of the Solid Spherical Harmonics and its Use in the Fast Multipole Methods. The Journal of Chemical Physics 104 (1996) 8003–8006.

[25] T. Pöschel, V. Buchholtz. Static Friction Phenomena in Granular Materials: Coulomb Law vs. Particle Geometry. Physical Review Letters 71, 24 (1993) 3963.

[26] T. Pöschel, S. Luding, eds. Granular Gases. Lecture Notes in Physics 564. Springer, Berlin, 2001.

[27] D.C. Rapaport. The Art of Molecular Dynamics Simulation. Cambridge University Press, Cambridge, 1995.

[28] V. Rokhlin. Rapid Solution of Integral Equations of Classical Potential Theory. Journal of Computational Physics 60 (1985) 187–207.

[29] S.A. Sauter. Variable Order Panel Clustering. Computing 64 (2000) 223–261.

[30] W. Schiehlen, P. Eberhard. Technische Dynamik. B.G. Teubner, Stuttgart, 2004.

[31] A. Schinner. Fast Algorithms for the Simulations of Polygonal Particles. Granular Matter 2, 1 (1999) 35–43.

[32] O. Steinbach. Numerische Näherungsverfahren für elliptische Randwertprobleme. Finite Elemente und Randelemente. B.G. Teubner, Stuttgart, 2003.

[33] J. Tausch. The Variable Order Fast Multipole Method for Boundary Integral Equations of the Second Kind. Computing 72 (2004) 267–291.

[34] P.A. Vermeer, S. Diebels, W. Ehlers, H.J. Herrmann, S. Luding, E. Ramm, eds. Continuous and Discontinuous Modelling of Cohesive Frictional Materials, Lecture Notes in Physics 568. Springer, Berlin, 2001.

[35] W. Wessel. Kontakterkennung räumlicher polyhedraler Körper mit Hilfe von Methoden der Molekulardynamik. Student-Thesis STUD-212, Institute of Engineering and Computational Mechanics, University of Stuttgart, Stuttgart, 2004.

[36] C.A. White, M. Head–Gordon. Derivation and Efficient Implementation of the Fast Multipole Method. The Journal of Chemical Physics 101 (1994) 6593–6605.

[37] P. Wriggers. Computational Contact Mechanics. John Wiley & Sons, Chichester, 2002.

[38] K. Yoshida. Applications of Fast Multipole Method to Boundary Integral Equation Method. Department of Global Environment Engineering, Kyoto University, Japan, 2001.

Erschienene Preprints ab Nummer 2005/1

2005/1	O. Steinbach	Numerische Mathematik 1. Vorlesungsskript.
2005/2	O. Steinbach	Technische Numerik. Vorlesungsskript.
2005/3	U. Langer G. Of O. Steinbach W. Zulehner	Inexact Fast Multipole Boundary Element Tearing and Interconnecting Methods
2005/4	U. Langer G. Of O. Steinbach W. Zulehner	Inexact Data–Sparse Boundary Element Tearing and Interconnecting Methods
2005/5	U. Langer O. Steinbach W. L. Wendland	Fast Boundary Element Methods in Industrial Applications Söllerhaus Workshop, 25.–28.9.2005, Book of Abstracts.
2005/6	U. Langer A. Pohoata O. Steinbach	Dual–Primal Boundary Element Tearing and Interconnecting Methods
2005/7	O. Steinbach (ed.)	Jahresbericht 2004/2005
2006/1	S. Engleder O. Steinbach	Modified Boundary Integral Formulations for the Helmholtz Equation
2006/2	O. Steinbach (ed.)	2nd Austrian Numerical Analysis Day, Book of Abstracts