Mechanism and Machine Theory 115 (2017) 77–96

Contents lists available at ScienceDirect

Mechanism and Machine Theory

journal homepage: www.elsevier.com/locate/mechmachtheory

Research paper

A benchmark study on accuracy-controlled distance calculation between superellipsoid and superovoid contact geometries

Artur Alves Gonçalves[a,b], Alexandre Bernardino[b,c], Joaquim Jorge[a,b], Daniel Simões Lopes[a,*]

[a] INESC-ID Lisboa, Portugal
[b] Instituto Superior Técnico, Universidade de Lisboa, Portugal
[c] ISR Lisboa, Portugal

ARTICLE INFO

Article history:
Received 10 February 2017
Revised 11 April 2017
Accepted 14 April 2017

Keywords:
Contact geometry
Ovoid shape
Superellipsoid
Implicit
Parametric
Mesh

ABSTRACT

Meshes are considered the gold standard regarding contact geometries of many mechanical models, even those represented with discrete surface contact elements. However, meshes may not be the best formulations when controlled precision and execution time become paramount. In this paper, we address parametric and implicit formulations for precise contact distance estimations between superovoidal shapes, which generalize superellipsoids. Parametric and implicit models provide more compact descriptions than meshes, while making it possible to approximate mechanical parts with great precision. Contrary to meshes, these geometric representations can then support fast calculation of distances with arbitrary precision without paying a storage or computation time penalty. We performed a benchmark study to compare different superellipsoidal and superovoidal contact geometry representations, including implicit surfaces, parametric surfaces and triangular meshes. We tested 10,000 contact pairs and also considered two application cases: robot fingers of an iCub and dental occlusion during bite. Our results show that the implicit model is the most efficient contact geometry representation, followed by parametric and mesh surfaces. In addition, results show that either implicit or parametric superovoids can provide more accurate distance estimations than meshes in practical settings where precise contact points, surface normals and clearance estimations are required.

1. Introduction

One of the most noteworthy interactions between bodies in a mechanical system is collision, i.e., when two bodies physically intersect at one point or region, exerting a force on one another [1,2]. Because collisions greatly influence the behavior of a mechanical system, they must be accurately modeled and accounted for. In Contact Analysis, minimum distance estimation is the operation used to compute whether two objects are intersecting, and if so, where that intersection occurs.

The minimum distance depends on the geometry of the considered objects. Therefore, the geometric representation of object should be chosen wisely, so that the simulated system is accurate enough when compared to its real counterpart. A

* Corresponding author. INESC-ID Lisboa, IST Taguspark, Room 2N9.1, Avenida Professor Cavaco Silva, 2744-016 Porto Salvo, Portugal.
E-mail addresses: artur.goncalves@tecnico.ulisboa.pt (A.A. Gonçalves), alexandre.bernardino@tecnico.ulisboa.pt (A. Bernardino), jorgej@acm.org (J. Jorge), daniel.lopes@inesc-id.pt, dsl.7125@gmail.com (D.S. Lopes).

http://dx.doi.org/10.1016/j.mechmachtheory.2017.04.008
0094-114X/

common approach is to approximate the objects by polyhedrons, also called meshes [3]. Since meshes can only represent straight edges and faces, any smooth curves must be discretized. This requires using a large number of vertices to achieve a faithful representation of the surface geometry. These high-resolution meshes lead to slow collision detection algorithms and high memory usage, which hinders their usability for real-time applications. In addition, mesh normals can be very noisy and change abruptly throughout the surface, due to its faceted nature.

Alternatively, objects can be defined using analytical curves and surfaces, such as boxes, spheres, and ellipsoids. In particular, superellipsoids (SE) [4] have been proposed, for which time- and memory-efficient minimum distance algorithms exist [5,6]. Superellipsoids have been used as contact surfaces for robotic haptic feedback devices [7,8], and there are examples of usage of these shapes to represent organic structures with high fidelity, such as human femoral heads [9]. Superellipsoids, together with spheres and ellipsoids, have been extensively studied in many collision detection applications. The superovoids, which generalize the superellipsoidal shape, appear as its natural successors. However, this model has not yet captured the attention of the contact analysis/detection communities. From a geometric modeling point of view, superovoids (SO) while marginally more complex than superellipsoids, can yield a higher accuracy in approximating shapes such as multi-fingered robotic hands, organic structures [9] and everyday life objects. Even the contact geometry of more complex objects with concavities can be approximated by multiple pairs of non-conformal superovoids arranged to fit concave surfaces in conformal contact configurations.

In this paper, we propose a flexible, mathematically well-defined, smooth convex surface called superovoid as a contact geometry model to be used in physical simulations. We present algorithms to compute the minimum distance between different contact geometry representations with (strictly) convex superellipsoidal and superovoidal shapes that interact in a non-conformal fashion. Note that by evaluating the minimum distance between smooth convex objects in a non-conformal configuration, then a quantitative figure reveals the collision state: distances greater than zero reveal that objects are apart; a null distance means that objects touch at a single point; and a "negative" distance represents objects that overlap. The scope of our work covers only non-conformal contact pairs (i.e., convex-convex), where the dimensions of the contact patch are both small compared to the curvatures, and also much smaller than the overall dimensions of the objects. Conformal contact pairs (i.e., convex-concave or concave-concave) are not considered in this paper.

Therefore, the focus of our paper is on the narrow-phase of the contact event and on the minimum distance calculation between (i) very close but still apart surfaces, (ii) single point contact or (iii) surfaces whose overlapping distance is much lesser than the smallest size parameter of a superellipsoid or superovoid. The goal is to determine the influence of contact geometry representation on closest distance calculation between superovoids and superellipsoids in a non-conformal configuration, during the narrow-phase of contact either apart or slightly overlapping. While seemingly minute, precise contact estimations are crucial to evaluating grip force and torques required to hold delicate objects without their breaking or falling to the ground.

We performed benchmark studies covering several aspects related to superovoidal contact geometry representation. In particular, we provide a comparison between implicit, parametric and mesh contact representations along with an extensive battery of tests. Note that comparison with other contact algorithms found in the literature is out of the scope of this paper.

Since meshes can model arbitrary geometries while superovoids cannot, to provide a fair comparison between contact geometries to address the tradeoff between accuracy and computational performance, we considered two case studies with complex objects, namely, robotic grasp motion planning and dental occlusion during chewing or rest, to test non-conformal (i.e., convex-convex) and conformal (i.e., convex-concave or concave-concave) configurations, respectively. As case studies to demonstrate the applicability of the proposed algorithms, we consider contact geometry models of (i) pairs of separated superovoids; (ii) pairs of overlapping superovoids; (iii) iCub robot fingers; and (iv) two molar teeth. We compare the superellipsoid and superovoid approach to an existing mesh-based implementation in terms of accuracy and computation time.

2. Literature review

The literature abounds on work related to collision detection between smooth convex surfaces, in particular, spheres, ellipsoids, and, in a lesser extent, superellipsoids. On the other hand, literature on (super) ovoid collision detection is scarce appearing, at most, in geometric modeling communities [9,10]. Since superovoids are natural extensions of (super) ellipsoids, this section covers recent work on contact detection between ellipsoidal and superellipsoidal shapes, including a couple of papers which served as inspiration to the proposed ovoidal contact algorithms [6,9].

2.1. Contact detection with smooth convex surfaces

Chen et al. 2012 [11] present a method to compute the minimum and maximum distance from a point to an ellipse or ellipsoid. The proposed method involves finding the root of one quartic equation to obtain the closest point to an ellipse, and a sequence of quartic equations for an ellipsoid. An iterative application of the method to find the minimum distance between two ellipsoids is presented, and achieves an error less than 10^{-6} in 11 iterations, but the method was not benchmarked or compared with other algorithms.

Pazouki et al. 2012 [12] propose an efficient and parallelizable method based on ellipsoids for simulating large scale multibody systems, instead of the traditional spheres. The algorithm is divided in three levels. On the lowermost level,

an unconstrained optimization is solved to find the closest points between two ellipsoids, based on the common normal concept method reported in [5], applied to ellipsoids. The intermediate level is a broad collision detection step, and detects if two ellipsoids are in contact or not, using sign of the roots of the characteristic equation of the pair of ellipsoids; this step is tailored for ellipsoids, as such an equation does not necessarily exist for any pair of surfaces. The third level is a space partitioning algorithm.

Collision detection can also relate to whether two objects are in contact or not without calculating any distance information, hence, act as a proximity query. An algorithm of this kind has been proposed by [13], which characterizes two ellipsoids into three distinct states: completely separated, externally touching, and overlapping. The algorithm involves computing the number of positive roots of a quartic equation, which is related to the state of the two ellipsoids. Instead of explicitly computing the roots, only five formulae are calculated, which take 28 multiplications and 12 additions. The authors then present a continuous collision detection algorithm using the mentioned function, and compare it to existing methods. The proposed algorithm is comparatively fast for degree 2 rigid-body and affine movements of the ellipsoids, but is an order of magnitude slower for movements represented by degree 4 transformations. It is, however, faster than the compared method if only the first time instant of collision is computed. Considering the experimental times for each collision detection – 0.1 to 0.5 ms for degree 2 movements – this algorithm may be adequate for real-time applications. It depends, however, on the usage of other algorithms to compute the exact point of collision between the ellipsoids, once they are asserted to be in contact.

Another proximity query is proposed by [14], in this case, an algorithm for detecting overlaps between two ellipsoids, in an exact manner, by calculating the roots of a convex function. As the method only detects collision, and not the point of contact, it could be used for elliptical bounding volumes, with a particularization for 3 dimensions, but other methods have to be employed to find the exact points of minimum distance.

Besides ellipsoids, the literature covers the usage of other smooth convex surfaces for the representation of contact geometries. For instance, contact between a superquadric surface and a 3D mesh can be detected by calculating the inside-outside function of the superquadric, at the relative position of each point of the mesh [15]. To avoid implementing edge-superquadric collision detection, and to improve contact resolution, additional points are inserted in the edges of each triangle of the mesh. The authors mention speed-ups of 20 to 30, comparing to regular mesh-mesh collision detections. Collision detection can also be performed based on distance fields: these are pre-calculated 3D arrays overlapped onto the considered mesh geometry, and store the distance from each cell to the mesh's surface. A more efficient variation proposed in [7] computes a bounding superellipsoid around the object's mesh and stores the distances to points in the surface of this superellipsoid instead. This reduces the distance field into a 2D array, saving a considerable amount of memory, but does not completely forgo the usage of polygonal meshes.

Also in the chemical engineering field, [16] propose discretizing superellipsoids into points (nodes) and perform collision detection by evaluating a surface's inside-outside function on the points defining another surface. The points are sorted according to their evaluation value between time-steps, so that they are not exhaustively tested every time-step. Two discretization approaches are studied which generate equally-spaced and curvature-adaptive points, respectively. This method is compared to a continuous contact detection algorithm based on the minimization of the sum of inside-outside functions of two surfaces, solved numerically. Performed experiments show that, while these methods are unsurprisingly slower than the usage of simple spheres, the behavior of the simulated system is significantly different. This suggests that simplification of non-spherical particles with spheres is not adequate for precision-critical simulations. Further comparisons with other contact geometry representations in [17] note that the discrete method is numerically more stable, but less precise and more memory-demanding, while the continuous inside-outside-based method is easier to implement but does not model the mechanical definition of contact as well as the common normal concept employed in [5,6].

Superellipsoids were extensively studied in [6,18,9] in the context of collision detection and shape modeling. The authors propose the usage of implicit smooth convex surfaces, namely superellipsoids, to model shapes of biomechanical structures [9], instead of using polygonal meshes or freeform surfaces. In [6] the authors proposed a new approach towards the reformulation of contact detection between convex implicit surfaces. Central to this reformulation are the implicit surface representation and the elegant way on how the Householder formula computes orthonormal sets given a normal surface vector [19]. Associated to this representation, a mathematical framework is presented to compute the minimum distance between two smooth convex surfaces, based on the common normal concept [6]. The Householder transformation is investigated for efficient and robust calculation of sets of orthogonal 3D vectors, and is then applied in said framework [19]. Finally, a numerical algorithm for the problem of finding the minimum distance between two superellipsoids is presented [6], as well as an analytical solution for the distance between a superellipsoid and a plane [18]. The developed work is applied in a case study related to simulation and analysis of human gait motion. Although the contact detection algorithm was formulated for any pair of smooth convex implicit surfaces, it has been specifically implemented for the (super)ellipsoid-(super)ellipsoid contact pair and the numerical results show that it has second-order convergence. However, this study lacks a comparative analysis between other contact geometry representations such as parametric surfaces and meshes.

2.2. Robot grasping

Contact geometries represented with smooth convex surfaces find useful applications in computational robotics. For robots to adequately interact with their environment, their actions should be planned with a sufficient degree of

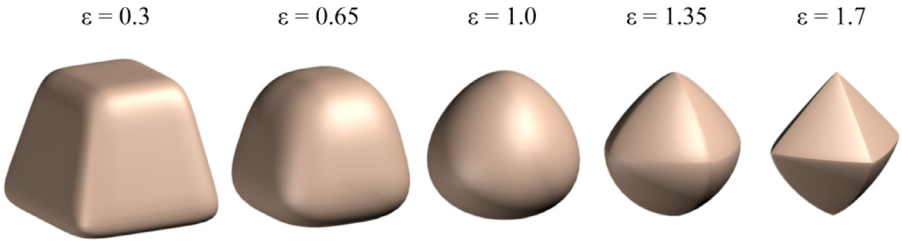

Fig. 1. Examples of superovoids for varying exponent values (with $\varepsilon_1 = \varepsilon_2$). The surface shapes are mediated between oval and squared frustum.

accuracy. The simulations for this planning involve, among other things, queries for the distance between objects and parts of the robot, and the penetration depth between intersecting objects and robot parts. Furthermore, these computations must run in real-time. In particular, the case study of this article focuses on the collision detection and minimum distance computations for grasp planning with the iCub robot hand [20]. Grasp planning techniques very often rely on the computation of grasp quality criteria, such as force closure [21], that require the position of the contact points and corresponding surface normals. Hence, the accurate computation of these quantities is highly important for the success of grasp planning algorithms.

As mentioned earlier, the adequate geometric representation of robotic parts is crucial for efficient collision detection. In [22], oriented bounding boxes and superquadrics are employed to represent a robotic hand and objects to be grasped. For a broad-phase planning, in [23] the volume in which the fingers can move is coarsely represented by cylinders. To exploit concave parts of objects, which provide a secure grip for grasping, [24] use (super)ellipsoids, elliptic cylinders and one-sheet hyperboloids.

The problem of representing humanoid hands appears in other areas as well: for hand pose estimation and tracking, in order to decrease the collision detection processing time, in [25], one hand is represented simply with overlapping spheres. Other more complex configurations exist: [26] employs spheres, cones and cylinders.

Collision detection also plays an important role in the simulation of complex biomechanical systems. The Simbody engine [27] was developed to accurately simulate the dynamics of biological structures, and is adjustable to real-time or high-fidelity simulations as needed. It supports rigid, deformable and elastic contact detection and handling, and is used in biomedical and prosthetic research, and in robotics, namely, in the Gazebo robot simulator [28].

Many specialized libraries exist to compute a series of collision-related queries, of which RAPID, SOLID, PQP and KCCD are examples, and focus on specific geometries such as Axis-Aligned Bounding Boxes (AABBs), Oriented Bounding Boxes (OBBs) and Continuous Collision Detection (CCD).

The KCCD library [29] specializes in real-time continuous collision detection for greater accuracy simulating very fast robotic movement, and computes minimum distances. Robot parts are represented with Sphere Swept Convex Hulls, avoiding the usage of mesh-based volumes. The library is able to detect collision and compute minimum distances between a pair or all pairs of robotic parts.

On the other hand, Flexible Collision Library (FCL) [30] performs a vast array of proximity queries, using rigid, deformable or articulated models, represented as primitive geometries, meshes, point clouds, and bounding volume hierarchies (BVH). FCL was designed to allow the addition of new collision detection methods, and different representations of objects. Out of the box, FCL supports spheres, boxes, cones, cylinders, capsules, planes and meshes. For collision detection purposes, meshes are represented with BVH. FCL is used in MoveIt! [31], a robotic motion planning, navigation and control library.

3. Superovoids

Central to this work is the adoption of superovoids as an idealized surface to represent contact geometry. Superovoids are capable of representing many shapes, both manmade and biological structures, and present the following desirable mathematical characteristics and properties: (i) have a limited and controllable set of parameters that affect global properties of the shape in a comprehensible manner; (ii) are symbolically represented by a compact and exact analytical representation which unequivocally describes the shape; (iii) are defined by real-valued continuous functions whose properties are well understood; and (iv) are smooth and convex, $C^n, n \geq 1, n \in \mathbb{N}$, continuous surfaces, and differentiable at any surface point, thus differential information (e.g., surface normals and curvatures) is available continuously over the surface, even where the surface is almost square or edgy.

From a geometric modeling perspective, superovoids are a natural extension of the superellipsoids presented by Barr [4]. In fact, the considered superovoids are tapered superellipsoids. Although superellipsoids are slightly more flexible than ellipsoids and spheres, meaning that they can model more varied objects, superovoids provide a greater convex modelling flexibility due to supraquadratic exponents and, most importantly, due to the introduction of a single axial asymmetry. But similar to superellipsoids, the surface shape varies between rounded and squared by varying the exponent values, being several examples shown in Fig. 1.

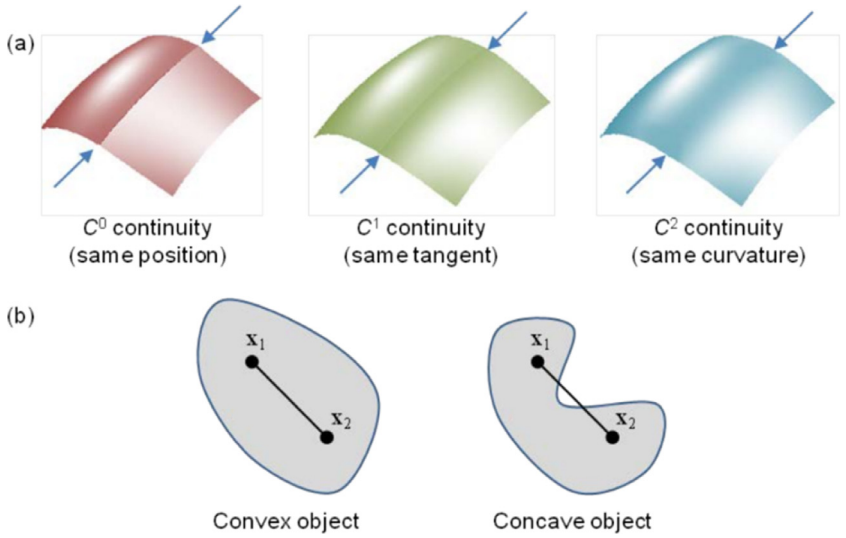

Fig. 2. (a) Smoothness or continuity on surface patches. (b) Convexity of objects.

3.1. Smooth convex geometries

Smoothness and convexity, visually depicted in Fig. 2, are two important geometrical properties for collision detection between superovoids. A surface is said to be smooth if it can be differentiated, up to a desired order, over its entire domain. Thus, smooth surfaces do not present pointy edges or spikes (i.e., singularities), and because they are differentiable at any point, the surface normals and tangents are always defined. By convention, surface normal vectors point outwards.

Regarding convexity, a 3-D object is said to be convex if for every pair of points within its geometric loci, then every point on the straight line segment that connects the pair of points is also within the geometric loci. That is, the following condition holds for every pair of points x_1, x_2 in the object, Ω

$$\mathbf{x}_3 = (1-\alpha)\mathbf{x}_1 + \alpha\mathbf{x}_2 \in \mathbf{\Omega}, \quad 0 < \alpha < 1 \tag{1}$$

where x_3 results from a linear combination of the two points on the object, which travels along a straight line segment as α increases from 0 to 1. Planes, cubes, spheres and ellipsoids are examples of other convex objects.

3.2. Implicit surface representation

Three-dimensional superovoidal shapes can be represented implicitly using an equation written as

$$F(x, y, z) = 0 \tag{2}$$

where x, y and z are the Cartesian coordinates of a 3D point located on the surface, and $F : \mathbb{R}^3 \to \mathbb{R}$ is a real valued function that defines, univocally, the shape surface. The implicit representation defines three distinct regions of space, here conventioned as follows: the points that satisfy (2) are on the surface; the set of points for which function evaluation is lower than zero are inside the surface, while those that yield a result greater than zero are outside the surface. Because of this property, $F$is also called an inside-outside function.

There are several representations of ovoidal shapes [10]. A very well described representation was proposed by [4], namely, the tapered superellipsoid. As the name suggests, superellipsoids are a generalization of the ellipsoid, and are defined by the following inside-outside function:

$$F_{SE}(x, y, z) = \left(\left(\frac{x}{a_1} \right)^{\frac{2}{\epsilon_1}} + \left(\frac{y}{a_2} \right)^{\frac{2}{\epsilon_1}} \right)^{\frac{\epsilon_1}{\epsilon_2}} + \left(\frac{z}{a_3} \right)^{\frac{2}{\epsilon_2}} - 1 \tag{3}$$

where a_1, a_2, $a_3 > 0$ are the lengths of the shape in the x, y and z axes, and $\epsilon_1, \epsilon_2 \in \ ]0, 2[$ are the squareness factors for the xOy plane and z axis, respectively. The range between 0 and 2 of the squareness exponents, ϵ_k, $k \in \{1, 2\}$, ensures that the resulting shapes are convex. Note that, to compute (3) and the following formulas presented in this paper with exponential terms, the exponentiations should be computationally implemented as

$$(x)^{\gamma} = (x)(x^2)^{\frac{\gamma-1}{2}}, \quad \forall x, \gamma \in \mathbb{R} \tag{4}$$

which has the correct order of evaluation to ensure that the final result is always a real number for any value of γ, otherwise a complex number would appear for $x < 0$.

By applying a tapering deformation on one of the superellipsoid's tip a superovoid is formed, which can be defined implicitly as

$$F_{SO}(x,y,z) = \left(\left(\frac{1}{T_x\frac{z}{a_3}+1}\right)^{\frac{2}{\epsilon_1}}\left(\frac{x}{a_1}\right)^{\frac{2}{\epsilon_1}} + \left(\frac{1}{T_y\frac{z}{a_3}+1}\right)^{\frac{2}{\epsilon_1}}\left(\frac{y}{a_2}\right)^{\frac{2}{\epsilon_1}}\right)^{\frac{\epsilon_1}{\epsilon_2}} + \left(\frac{z}{a_3}\right)^{\frac{2}{\epsilon_2}} \tag{5}$$

where and $T_x, T_y \in [-0.5,\ 0.5]$ are the tapering (egg-shape) factors in x and y, respectively. Note that the factors that multiply the x and y coordinates depend on z, so when $T_x = T_y = 0$, the superovoid becomes symmetrical in relation to its equatorial plane (it does not have an ovoidal shape anymore), becoming a superellipsoid [4].

The surface normal direction, $\mathbf{n}_{SO} = \nabla F = [\mathrm{n_x}\ \mathrm{n_y}\ \mathrm{n_z}]^{\mathrm{T}}$, can be readily computed by taking the gradient of F_{SO}, which is a useful property for computing the minimum distance between surfaces:

$$\begin{aligned}
n_x(x,y,z) &= \frac{2}{a_1\varepsilon_2}\left(\frac{1}{T_x\frac{z}{a_3}+1}\right)^{\frac{2}{\epsilon_1}}\left(\frac{x}{a_1}\right)^{\frac{2}{\epsilon_1}-1}\lambda(x,y,z)\\
n_y(x,y,z) &= \frac{2}{a_2\varepsilon_2}\left(\frac{1}{T_y\frac{z}{a_3}+1}\right)^{\frac{2}{\epsilon_1}}\left(\frac{y}{a_2}\right)^{\frac{2}{\epsilon_1}-1}\lambda(x,y,z)\\
n_x(x,y,z) &= \frac{2}{a_3\varepsilon_2}\left(-\left(T_x\left(T_x\frac{z}{a_3}+1\right)^{-\frac{2}{\epsilon_1}-1}\left(\frac{x}{a_1}\right)^{\frac{2}{\epsilon_1}}\right)\lambda(x,y,z)\right.\\
&\quad\left.-\left(T_y\left(T_y\frac{z}{a_3}+1\right)^{-\frac{2}{\epsilon_1}-1}\left(\frac{y}{a_2}\right)^{\frac{2}{\epsilon_1}}\right)\lambda(x,y,z) + \left(\frac{z}{a_3}\right)^{\frac{2}{\epsilon_2}-1}\right)
\end{aligned} \tag{6}$$

with

$$\lambda(x,y,z) = \left(\left(\frac{1}{T_x\frac{z}{a_3}+1}\right)^{\frac{2}{\epsilon_1}}\left(\frac{x}{a_1}\right)^{\frac{2}{\epsilon_1}} + \left(\frac{1}{T_y\frac{z}{a_3}+1}\right)^{\frac{2}{\epsilon_1}}\left(\frac{y}{a_2}\right)^{\frac{2}{\epsilon_1}}\right)^{\frac{\epsilon_1}{\epsilon_2}-1} \tag{7}$$

3.3. Parametric surface representation

Superovoids can also be described using an angle-center parametrization. Parametric representation is based on a mapping between the surface in 3D space and a 2D parameter domain, written in the form

$$\mathbf{s}(u,v) = \begin{bmatrix} x = x(u,v) \\ y = y(u,v) \\ z = z(u,v) \end{bmatrix}, \quad \begin{matrix} u_1 < u < u_2 \\ v_1 < v < v_2 \end{matrix} \tag{8}$$

where $s : \mathbb{R}^2 \to \mathbb{R}^3$ is the parametric surface function, and parameters u and v generate the whole set of points on the surface, when swept over the domain $[u_1, v_1] \times [u_2, v_2] \in \mathbb{R}^2$. Contrary to the implicit representation, generating polygonal meshes from parametric surfaces is straightforward, and this representation is frequently used in computational geometry and design applications [32,33]. By changing the parametric coordinates, the same shape can be defined by multiple parametric representations.

To represent a superovoid described as a tapered superellipsoid, it is necessary to use the angle-center parametrization of a superellipsoid [5]

$$\mathbf{s}_{SE}(\varphi_1,\varphi_2) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_1(\cos\varphi_1)^{\epsilon_1}(\cos\varphi_2)^{\epsilon_2} \\ a_2(\sin\varphi_1)^{\epsilon_1}(\cos\varphi_2)^{\epsilon_2} \\ a_3(\sin\varphi_2)^{\epsilon_2} \end{bmatrix}, \quad \begin{matrix} -\pi < \varphi_1 < \pi \\ -\frac{\pi}{2} < \varphi_2 < \frac{\pi}{2} \end{matrix} \tag{9}$$

where exponentiations are implemented as described in (4), and the x and y coordinates are multiplied by a factor varying linearly with z:

$$\begin{aligned}
\mathbf{s}_{SO}(\varphi_1,\varphi_2) &= \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_1\left(T_x\frac{z}{a_3}+1\right)(\cos\varphi_1)^{\epsilon_1}(\cos\varphi_2)^{\epsilon_2} \\ a_2\left(T_y\frac{z}{a_3}+1\right)(\sin\varphi_1)^{\epsilon_1}(\cos\varphi_2)^{\epsilon_2} \\ a_3(\sin\varphi_2)^{\epsilon_2} \end{bmatrix}\\
&= \begin{bmatrix} a_1\left(T_x(\sin\varphi_2)^{\epsilon_2}+1\right)(\cos\varphi_1)^{\epsilon_1}(\cos\varphi_2)^{\epsilon_2} \\ a_2\left(T_y(\sin\varphi_2)^{\epsilon_2}+1\right)(\sin\varphi_1)^{\epsilon_1}(\cos\varphi_2)^{\epsilon_2} \\ a_3(\sin\varphi_2)^{\epsilon_2} \end{bmatrix}, \quad \begin{matrix} -\pi < \varphi_1 < \pi \\ -\frac{\pi}{2} < \varphi_2 < \frac{\pi}{2} \end{matrix}
\end{aligned} \tag{10}$$

while the tangent surface vectors are given by differentiating Eq. (10) in order to φ_1 and φ_2, respectively:

$$\mathbf{t}_{SO}(\varphi_1,\varphi_2)=\begin{bmatrix}-a_1\varepsilon_1\left(T_x(s\varphi_2)^{\epsilon_2}+1\right)(c\varphi_2)^{\epsilon_2}(s\varphi_1)(c\varphi_1)^{\epsilon_1-1}\\ -a_2\varepsilon_1\left(T_y(s\varphi_2)^{\epsilon_2}+1\right)(c\varphi_2)^{\epsilon_2}(c\varphi_1)(s\varphi_1)^{\epsilon_1-1}0\end{bmatrix},\quad \begin{matrix}-\pi<\varphi_1<\pi\\ -\frac{\pi}{2}<\varphi_2<\frac{\pi}{2}\end{matrix} \tag{11}$$

$$\mathbf{b}_{SO}(\varphi_1,\varphi_2)=\begin{bmatrix}a_1\varepsilon_2(c\varphi_1)^{\epsilon_1}\left(T_x(c\varphi_2)(s\varphi_2)^{\epsilon_2-1}(c\varphi_2)^{\epsilon_2}-\left(T_x(s\varphi_2)^{\epsilon_2}+1\right)(s\varphi_2)(c\varphi_2)^{\epsilon_2-1}\right)\\ a_2\varepsilon_2(s\varphi_1)^{\epsilon_1}\left(T_y(c\varphi_2)(s\varphi_2)^{\epsilon_2-1}(c\varphi_2)^{\epsilon_2}-\left(T_y(s\varphi_2)^{\epsilon_2}+1\right)(s\varphi_2)(c\varphi_2)^{\epsilon_2-1}\right)\\ a_3\varepsilon_2(c\varphi_2)(s\varphi_2)^{\epsilon_2-1}\end{bmatrix},\quad \begin{matrix}-\pi<\varphi_1<\pi\\ -\frac{\pi}{2}<\varphi_2<\frac{\pi}{2}\end{matrix} \tag{12}$$

where $s\varphi=\sin(\varphi)$ and $c\varphi=\cos(\varphi)$, while $\mathbf{n}_{SO}$ is obtained by the cross product between $\mathbf{t}_{SO}$ and $\mathbf{b}_{SO}$.

3.4. Polygonal representation

As for the polygonal mesh representation of a superovoid, it consists of a collection of polygons, stored as the list of vertices that define a facet. Typically, meshes are composed of triangles, as 3 vertices forming a triangle are guaranteed to be coplanar, and any polygonal shape can be decomposed into triangles. Triangular meshes are the gold standard for 3D modeling and visualization, as they can approximate almost any kind of object, including rigid and deformable bodies. However, meshes can only faithfully represent faceted objects; smooth surfaces must be approximated by triangulation, and more accuracy is only achieved by using a greater number of vertices and triangles, leading to a higher memory usage, and slower collision detection algorithms. Furthermore, mesh edges are only C^0-continuous, which means that the surface normal direction is not defined in these regions.

3.5. Hierarchical representations

Hierarchical representations are spatial data structures that are used as acceleration structures for collision detection. Essentially, they consist of a hierarchical grouping of 3D objects where each group is associated with a conservative bounding geometric primitive, hence, are nested structures of recursive nature.

Through a hierarchal representation, complex objects are modeled using multiple geometric primitives, each with a corresponding to a proximity query. These proximity queries may not always result in useful computational work, as the bodies might not collide during a significant part of the simulation. For this reason, it may be efficient to approximate each complex object by simpler geometric primitives which fully enclose the object. This is also a form of broad collision detection, and the geometric primitives are called bounding volumes. The objective is to first execute the collision detection between these bounding volumes, which are faster because of the simplified geometry of the bounding volumes. If these do not indicate a collision between the bounding volumes, it is certain that the two objects are not colliding. Therefore, no more processing is needed regarding those two objects. Bounding volume hierarchies can be employed to boost efficiency: the levels of the hierarchy are composed of an increasing number of bounding volumes, fitting the considered object more and more tightly.

There are many well-studied and commonly-used bounding volumes (van den Bergen 1997), for which there are simple and efficient collision tests such as sphere-like and box-like shapes.

For instance, spheres are widely used as collision bounding volumes, as the problem of finding the minimum distance between two spheres has a simple closed-form solution: two spheres are intersecting if the Euclidean distance between their centers is lower than the sum of the radii of the spheres. Another commonly used shape are capsules, which are the union of a cylinder with two spheres on top, all with the same radius. They can also be seen as the swept volume taken by a sphere as it moves in a straight line segment. Collision between capsules in arbitrary poses can always be calculated using the distance between their central axis segment, and comparing it to the sum of their radii.

Box-like shapes are also very familiar in collision detection. Axis-aligned bounding boxes (AABB) are boxes whose edges are all collinear with one of the Cartesian coordinate axes, X, Y and Z. Collision detection is trivial with these boxes, by interpreting each dimension as the interval of coordinates in which the box exists. However, if the enclosed object rotates, the AABB must be recalculated to avoid part of the object from leaking through the box. Oriented bounding boxes (OBB) are a generalization of AABB that can be rotated with the object as it moves, providing a tighter fit and avoiding the constant recalculation of the box. However, the collision detection is computationally heavier.

Polygonal bounding volumes are interesting alternatives to the sphere-like and box-like counterparts. Discrete oriented polytopes (DOP) are another generalization of the AABB: they are composed of K faces (K-DOP), and are built by considering a number of appropriately-oriented planes far away from the object, which are progressively advanced until they collide with the object. The bounding volume is the intersection of the half-spaces bounded by these planes, and is always a convex polytope. With 6 planes perpendicular to each axis, the 6-DOP degenerates into a simple AABB. With 18 planes, all the edges of an AABB can be beveled, creating a tighter fit around the object. With 26 planes, all the corners can also be beveled. K-DOPs can be seen as a particularization of a convex bounding mesh, and the collision detection is increasingly more expensive to compute with higher number of planes.

Since our focus is on the narrow-phase of the contact event and on minimum distance calculation, we do not consider our superovoid approach as a proximity query as more efficient methods exist to determine if two surfaces are far apart

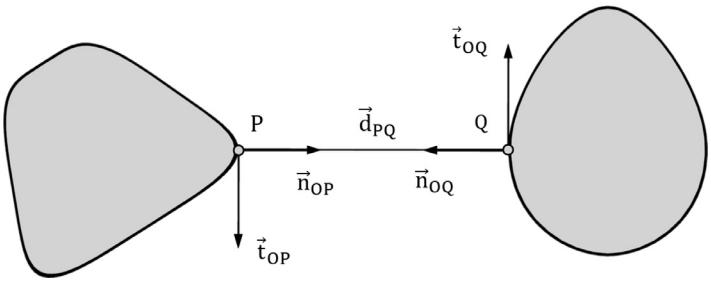

Fig. 3. Orthogonal and collinear vector relationships that define the common normal concept among the surface normals, the distance vector, and the tangent vectors. Surfaces are represented in 2D, thus, the bitangent vectors are not shown. By convention, surface normal vectors point outwards.

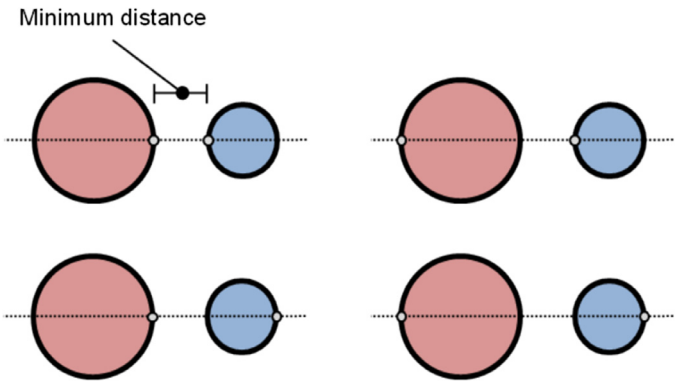

Fig. 4. A pair of spheres illustrating that the common normal concept is only a necessary condition for determining the minimum distance between smooth convex surfaces.

(e.g., AABB, OOBB, Bounding Spheres, etc). Therefore, to ensure rapid computations, a hierarchy of AABBs was considered as the acceleration structure for the meshless approach, i.e., each meshed superovoid was bounded by an axis aligned box. Whenever the proximity query was true, then the narrow-phase method was executed to compute the minimum distance points between contact surface pairs.

4. Contact algorithms for superovoids

The proposed algorithms are based on [6] and can compute the minimum distance between combinations of (strictly) convex superellipsoids and superovoids that interact in a non-conformal fashion, using a numerical iterative approach to solve the equations provided by the common normal concept. Hence, such algorithms answer the following problem: given two smooth convex surfaces in 3D space, what is the pair of surface points that defines that minimum distance between them? Using the distance between these points, it is possible to assert whether the surfaces are in collision or not.

4.1. Common normal concept

The common-normal concept [2]states an important condition for the solution of this problem: let P and Q be two points on two C^1 continuous surfaces (i) and (j), respectively, which are in contact or have a minimum distance expressed by those points; if the normal directions of each surface point, $\mathbf{n}_{OP}$ and $\mathbf{n}_{OQ}$, are the same and equal to the direction of the distance vector $\mathbf{d}_{PQ}$, that links them, then points P and Q embody the common-normal concept. As a consequence, the tangent and bitangent vector directions on both surface points, $\mathbf{t}_{OP}$, $\mathbf{t}_{OQ}$, $\mathbf{b}_{OP}$, $\mathbf{b}_{OQ}$, define two planes parallel to each other. A 2D case is illustrated in Fig. 3.

Note that the common normal concept outlines a necessary condition for finding the minimum distance between two surfaces. However, it is not a sufficient condition: the fact that two points in the surfaces share the same normal does not imply that they define the minimum distance between the surfaces. The example with spheres in Fig. 4 illustrates this: while there are 4 potential pairs of points with shared normal direction, only one pair corresponds to the minimum distance.

The distance vector magnitude is calculated as the signed Euclidean distance of the vector formed by the surface points. Note that a signed distance can be a positive, null, and negative value. Hence, at a given time instant, the signed magnitude of the distance vector indicates one of the three possible contact situations: (i) if the surfaces intersect at a single point, the minimum distance is zero valued; (ii) if the surfaces are not contacting, this distance is positive; and (iii) for overlapping surfaces, this distance is considered negative.

4.2. Implicit representation

The common normal condition requires mathematical expressions of normal and tangential surface vectors. For implicit surfaces, the expression for the normal vector to the superovoid surface can be easily obtained by taking the gradient of the inside-outside function (6), while tangent and bitangent directions can only be obtained using a vector orthogonalization technique. For this purpose, the Householder transformation is used, as it provides a well-defined analytical solution to compute the tangent and bitangent vector spaces of implicit surfaces [19]. Compared to other vector orthogonalization techniques, the Householder formula is more efficient to calculate both a locally and globally consist of tangent and bitangent vector fields by only giving the surface normal. This transformation is expressed as

$$\mathbf{H} = \mathbf{I} - 2\frac{\mathbf{h}\mathbf{h}^T}{\mathbf{h}^T\mathbf{h}} = \begin{bmatrix} 1-2\frac{h_1^2}{h^2} & -2\frac{h_1h_2}{h^2} & -2\frac{h_1h_3}{h^2} \\ -2\frac{h_1h_2}{h^2} & 1-2\frac{2h_2^2}{h^2} & -2\frac{h_2h_3}{h^2} \\ -2\frac{h_1h_3}{h^2} & -2\frac{h_2h_3}{h^2} & 1-2\frac{h_3^2}{h^2} \end{bmatrix} \tag{13}$$

where

$$\mathbf{h} \equiv \mathbf{h}(\mathbf{n}) = \begin{bmatrix} h_1 & h_2 & h_3 \end{bmatrix}^T = \begin{bmatrix} \{h_1 \in \{n_x - \|\mathbf{n}\|_2,\ n_x + \|\mathbf{n}\|_2\} | h_1 \notin \{0\}\} & n_y & n_z \end{bmatrix}^T \tag{14}$$

where $\mathbf{n} = [n_x\ n_y\ n_z]^{\mathrm{T}}$ is the given normal vector, $n \equiv \|\mathbf{n}\|_2$ and $h \equiv \|\mathbf{h}\|_2$ are the Euclidean norm of the normal vector $\mathbf{n}$ and auxiliary vector $\mathbf{h}$, respectively. In order to prevent $h_1 = 0$ ($h_1 = 0$ nullifies the first row column of H which becomes singular), h_1 results from the relative complement between sets $\{n_x - \|\mathbf{n}\|^2, n_x + \|\mathbf{n}\|^2\} \backslash \{0\}$.

Matrix $\mathbf{H}$ is symmetric and orthogonal, and its rows (or columns) form an orthogonal vector basis. Its first column is collinear to $\mathbf{n}$. Its second and third columns are perpendicular to $\mathbf{n}$, and can thus be considered as tangent $\mathbf{t}(\mathbf{n})$ and bitangent $\mathbf{b}(\mathbf{n})$ directions associated with the normal at a given surface point.

The common normal concept for implicit surfaces can be mathematically expressed as the following system of non-linear equations:

$$\mathbf{\Phi}(\mathbf{q}) = \mathbf{0} \Leftrightarrow \begin{bmatrix} \mathbf{n}_{OP}^T \cdot \mathbf{t}_{OQ} \\ \mathbf{n}_{OP}^T \cdot \mathbf{b}_{OQ} \\ \mathbf{d}_{PQ}^T \cdot \mathbf{t}_{OQ} \\ \mathbf{d}_{PQ}^T \cdot \mathbf{b}_{OQ} \\ F_i \\ F_j \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{15}$$

where $\mathbf{\Phi}$ is the vector of geometric constraints, $\mathbf{\Phi} : \mathbb{R}^6 \rightarrow \mathbb{R}^6$, and

$$\mathbf{q} \equiv \mathbf{q}_{implicit} = \begin{bmatrix} x_i & y_i & z_i & x_j & y_j & z_j \end{bmatrix}^T \tag{16}$$

is the vector of unknowns, containing the Cartesian coordinates of the potential minimum distance points, in the local coordinate system of each implicit surface. The last two geometric constraints in (15) are geometric loci constraints, and ensure the points are on the surface of the superovoids.

For the case of superellipsoids and superovoids, the vector of geometric constraints $\mathbf{\Phi}$ is composed of C^2 continuous functions, which makes the function twice-differentiable. As such, (15) can be solved using the iterative Newton-Raphson method. An improved approximation to the solution (16) $\mathbf{q}_{k+1}$ is computed based on a potential candidate $\mathbf{q}_k$:

$$\mathbf{q}_{k+1} = \mathbf{q}_k - (\mathbf{\Phi}_{\mathbf{q}}(\mathbf{q}_k))^{-1}\mathbf{\Phi}(\mathbf{q}_k) \tag{17}$$

where $\mathbf{\Phi}_{\mathbf{q}}$ is the Jacobian matrix of the geometric constraints function (15) and k is the Newton-Raphson iteration index. This Jacobian matrix is computed numerically using finite differences. For performance reasons, the Jacobian matrix inversion is not explicitly computed; instead, the linear system $(\mathbf{\Phi}_{\mathbf{q}}(\mathbf{q}_k))x = \mathbf{\Phi}(\mathbf{q}_k)$ is solved for x and this x is then subtracted from $\mathbf{q}_k$ to obtain $\mathbf{q}_{k+1}$.

The iterative procedure stops once the difference between $\mathbf{q}_{k+1}$ and $\mathbf{q}_k$ is less than a certain tolerance, which can be easily adjusted to compute more accurate results, or accelerate the algorithm at the cost of precision. Here, the Newton-Raphson procedure stops once the tolerance of 10^{-6} is reached, or when it reaches 30 iterations.

4.3. Parametric representation

By definition, the parametric representation guarantees that a given point in the parameter space always lies on the surface. Therefore, the last two conditions in (15) are not considered for the parametric case, as the first four conditions are sufficient to find the four unknowns. Thus, the vector of geometric constraints for parametric surfaces becomes

$$\mathbf{\Phi}(\mathbf{q}) = \mathbf{0} \Leftrightarrow \begin{bmatrix} \mathbf{n}_{OP}^T \cdot \mathbf{t}_{OQ} \\ \mathbf{n}_{OP}^T \cdot \mathbf{b}_{OQ} \\ \mathbf{d}_{PQ}^T \cdot \mathbf{t}_{OQ} \\ \mathbf{d}_{PQ}^T \cdot \mathbf{b}_{OQ} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{18}$$

and the vector of unknowns is written as

$$\mathbf{q} \equiv \mathbf{q}_{parametric} = \begin{bmatrix} \varphi_{1i} & \varphi_{2i} & \varphi_{1j} & \varphi_{2j} \end{bmatrix}^T \tag{19}$$

where $\varphi_{1,\,2i,\,j}$ are the parametric coordinates of two potential solution points for the minimum distance problem, on surfaces i and j.

4.4. Initial iteration

The presented algorithm uses numerical methods. As such, an initial guess is needed to iterate over and reach an approximation to the solution. Due to the local nature of the Newton-Raphson method, a poor guess of the initial candidate points may lead the algorithm either to diverge or to perform unnecessary iterations until reaching a proper solution. The algorithm thus requires a robust technique to provide an initial guess close to the desired solution.

To this end, we explored two techniques to find acceptable initial guesses, one for each surface representation. For implicit surfaces, an octree approximation of each superovoid is generated. To build the octrees, the inside-outside function of the superovoid is evaluated at the center of each octree cell: the cells whose value is lower than 0 are considered "filled". The minimum distance between a pair of octrees is computed with FCL by recursively evaluating the distances at each hierarchical level of the octree. If the distance found for a pair of cells is lower than the minimum distance found until that point, the algorithm recurs on those branches of each octree. Each leaf cell is represented by a box with the same orientation as the original superovoid. By the end of the recursion, the algorithm will have found the two leaf-cells that define the minimum distance between the two octrees. These minimum distance points between the two octrees are then used as an initial guess for the Newton-Raphson algorithm (17). As for meshes, the FCL code already supports octree minimum distance calculations on octrees represented with OctoMap [34].

For parametric surfaces, the two-dimensional parametric space of each superovoid (9) is discretized into a quadtree, and the center of each cell is considered a candidate point. For each level of the quadtree, every pair of candidate points formed between both superovoids is evaluated (16 pairs) and their Euclidean distances computed. The algorithm then selects the two cells for which the computed distance is the smallest, and repeats the process inside the two selected cells for a given number of iterations. Experiments suggest that 6 iterations yield acceptable estimations and allow the numerical procedure to converge. Finally, the two parametric coordinates found in the final iteration are used as the initial estimation for the parametric Newton-Raphson procedure.

Since the relative position of two moving surfaces is likely not to change significantly during short time-steps (i.e., assuming that there is temporal coherence), the minimum distance points computed in one collision query are used as the initial candidate points in the following collision query.

4.5. Accuracy and efficiency metrics

The implicit and parametric minimum distance methods proposed in this paper were compared with the existing FCL mesh-based method, using the geometrical accuracy and computational efficiency metrics exposed in this subsection.

The accuracy of the proposed superellipsoid and superovoid algorithms is enforced by the Newton-Raphson procedure, which takes a tolerance as a parameter. This is the tolerance for the Euclidean norm between two consecutive iterations $\mathbf{q}_k$ and $\mathbf{q}_{k+1}$ (16), which we take as a bound for the error between the last iteration and the true solution. This last iteration, the solution found by our proposed algorithms, will be referred to as $\mathbf{q}_{SO}$.

The accuracy of the mesh version was defined as

$$accuracy = |\mathbf{q}_{mesh} - \mathbf{q}_{SO}| \tag{20}$$

where $\mathbf{q}_{mesh}$ is the 6×1 vector containing the coordinates of the two minimum distance points found by the mesh algorithm, with the same layout as $\mathbf{q}_{SO}$. The solution found by the superovoid algorithms is considered to be the "ground truth", because Newton-Raphson enforces a tolerance of 10^{-6}, which is much lower than $|\mathbf{q}_{mesh} - \mathbf{q}_{SO}|$ for the obtained results, shown in Section 6.

The computational efficiency of the algorithms is measured as the necessary time to find the minimum distance points between the two surfaces. For the proposed algorithms, this time includes finding the initial iteration for the Newton-Raphson procedure. For the mesh algorithm, this does not include generating the mesh vertices and triangles. Collision queries which are interrupted due to broad-phase culling were not considered.

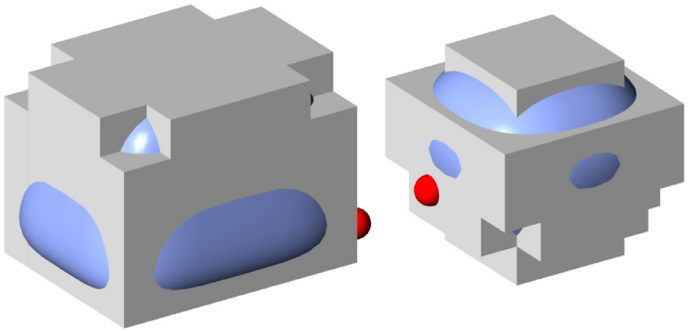

Fig. 5. Example of the estimation of the initial candidate points for implicit superovoids: the illustration shows the octrees (light grey) overlaid onto the respective superovoids and the minimum distance points between the two octrees (red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

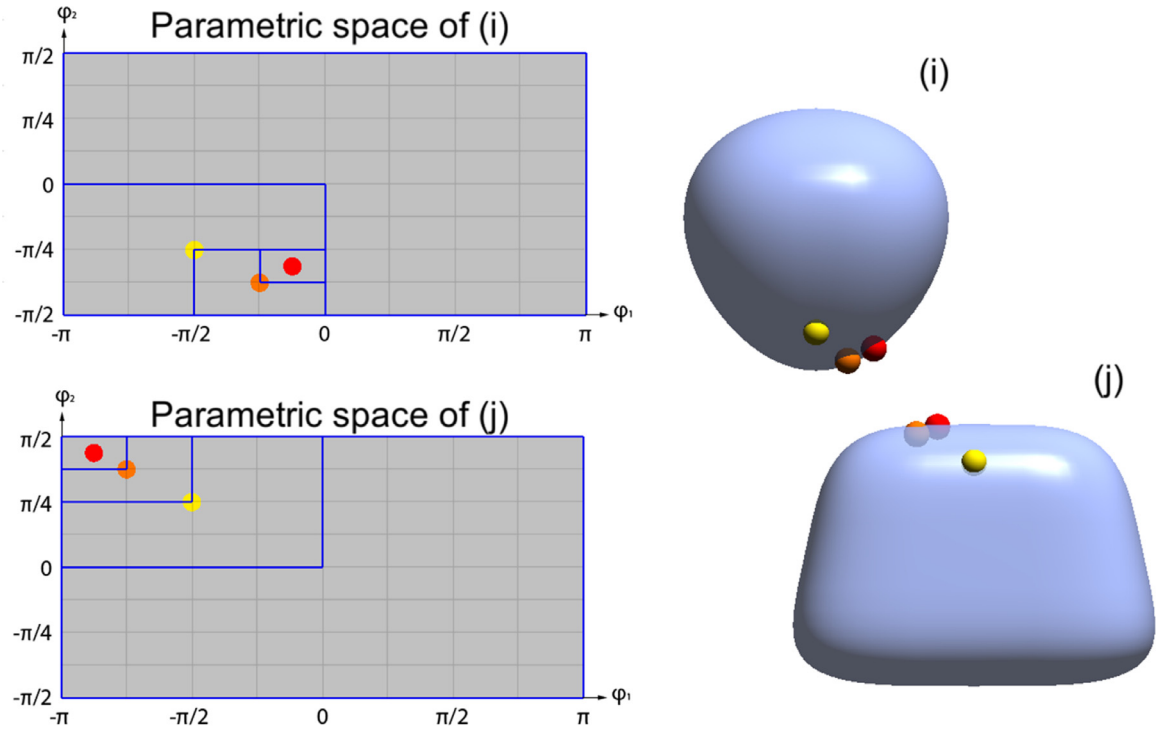

Fig. 6. Example of the estimation of the initial candidate points for parametric superovoids: execution of the quadtree method (3 iterations). On the left, the parametric spaces of each surface, along with the quadtree division and the iterations selected by the algorithm. Yellow is the first iteration, red is the last. On the right, a 3D representation of the surfaces, their relative position, and the corresponding 3D points of the iterations, in matching colors. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.6. Implementation

The superovoid shape and the described algorithm were implemented in FCL, in C++. The matrix operations necessary for the Newton-Raphson method use the LAPACK library [35]. The code has been published as a fork of FCL's repository, free and open source. Slight modifications were introduced in ROS [36] and MoveIt! [31] to support this new shape. Namely, the URDF parsing modules were extended with a new entity called "SUPEROVOID", characterized by 7 parameters, and an identifier for "SUPEROVOID" was added to several "enums" representing the supported surfaces.

A visualization development tool was created with Unity3D, using our modified version of FCL as a dynamic library, which allowed us to visually confirm and easily showcase the implemented surfaces and algorithms, and their outputs (minimum distance points, normals, tangents). This tool was also used to visualize the mesh-based algorithm that FCL already implements. Several figures, including Fig. 5, Fig. 12, and the graphical abstract, were produced with this visualization tool.

5. Case studies

Several case studies were considered to benchmark the minimum distance algorithms for each contact geometry type. A total of five case studies are tested: (i) battery of ten thousand overlapping surfaces; (ii) battery of ten thousand separated surfaces; (iii) single contact pair of superovoids to address geometric accuracy; (iv) iCub hand representation; and (v) dental occlusion. The algorithms under analysis follow one of the following schemes: octImplicit (octree approximation and numerical resolution of Eq. (15)), quadParametric (quadtree approximation and numerical resolution of Eq. (18)), quadImplicit (quadtree approximation and numerical resolution of Eq. (15)), ntbParametric (quadtree approximation and numerical resolution of Eq. (18) with implicit normal, tangent and bi-tangent vectors), or triangular mesh approach.

5.1. Randomized battery of tests

To further validate and benchmark our proposed algorithms, an extensive battery of tests was performed, comprising of 10,000 minimum distance queries for each of the compared algorithms. Each test consisted in computing the minimum distance between two superovoids, placed at coordinates (0, 0, 0) and (0, 2.21, 0), with fixed sizes $a_1 = a_2 = a_3 = 1.0$ and uniformly distributed random orientations and parameters in the ranges, $\varepsilon_1, \varepsilon_2 \in [0.3, 1.1]$, $T_x, T_y \in [-0.4, 0.4]$. These parameters lead to situations in which the two superovoids are either separated, or slightly overlapped. The two situations were studied separately.

The minimum distance between the two superovoids was computed using four different approaches: (i) quadImplicit scheme which relies on the quadtree initial iteration and the implicit algorithm described in Section 4.2; (ii) quadParametric scheme which considers the quadtree initial iteration and uses the parametric algorithm described in Section 4.3; (iii) ntbParametric that is also uses the parametric algorithm from Section 4.3 but with normals, tangents and bitangents are computed using the implicit formulas instead of the parametric ones; (iv) FCL's mesh-based minimum distance algorithm. For the latter, mesh approximations of superovoids of 290 vertices (576 triangles) were used. The initial iteration for the Newton-Raphson algorithm was the same for cases (i), (ii) and (iii), and was obtained with the parametric-quadtree method described in Section 4.4 using 6 iterations.

5.2. iCub Hand representation

As a case study to verify the performance of the different contact geometry representations of a superovoid, we consider the minimum distance calculation for robotic grasp motion planning. To perform the tests, the iCub robot's hand (see Fig. 7) [20] was represented with superovoids and superellipsoids, and comparisons were made with an existing mesh-based model in terms of accuracy and computation time of the minimum distance algorithms. The algorithm described in Section 5 was implemented in FCL, and has been published as a fork of its online repository, free and open-source (https://github.com/artur-ag/fcl). The presented robotics case-study was completed in the MoveIt! [31] motion planning software, which only performs geometrical collision detection. The adjustable accuracy of smooth convex contact geometries will surely be of even greater value if applied to physics-based tools such as Gazebo [28], which simulate contact forces, torques and friction for robotic applications.

The iCub robot's hands were manually approximated with superovoids and superellipsoids, using the original CAD drawings as reference. An approximation using polygonal meshes, available at [37] as a URDF file, was used for comparison. The superovoid and superellipsoid versions are implemented as modifications of this URDF file. The 5 fingers are represented with a total of 21 surfaces. Each individual link is around 16 mm long.

Special attention was given to the underside and tips of the iCub's fingers, as these are the part of the surfaces that come in contact with objects during grasping interactions. As such, parts of the proposed geometries cover empty space on top of the fingers, but this is not problematic for the application in mind.

The simulation consisted on moving the iCub's right hand towards its left, palms facing each other, until the tips of both thumbs touched, and then rotating the right hand inward, causing the fingers of the right hand to intersect and pass through the fingers of the left. This ensured plenty of collisions between the different links of the fingers. The movements were manually performed in the RViz interface. The proposed octImplicit and quadParametric schemes were timed and compared with FCL's mesh minimum distance query. The queries were timed only when the considered pair of surfaces were intersecting.

The following hand representations were considered, with mesh, octImplicit and quadParametric schemes tested separately: (i) the simplified triangular meshes of the iCub hand from [37] using the existing FCL BVH proximity query; (ii) the simplified triangular meshes of the iCub hand from [37] using the existing FCL minimum distance query; (iii) octImplicit superellipsoids (Fig. 7(b)) using the superellipsoid-specific implementation, which is mathematically simpler than for superovoids; (iv) octImplicit superovoids (Fig. 7(b)) but with the general superovoid implementation; (v) octImplicit mixed representation (Fig. 7(c)) using the superovoid implementation for the fingertips and the simpler superellipsoid implementation for the other links; (vi) quadParametric superellipsoids (Fig. 7(b)) using the superellipsoid-specific implementation, which is mathematically simpler than for superovoids; (vii) quadParametric superovoids (Fig. 7(b)) but with the general superovoid implementation; (viii) quadParametric mixed representation (Fig. 7(c)) using the superovoid implementation for the fingertips and the simpler superellipsoid implementation for the other links.

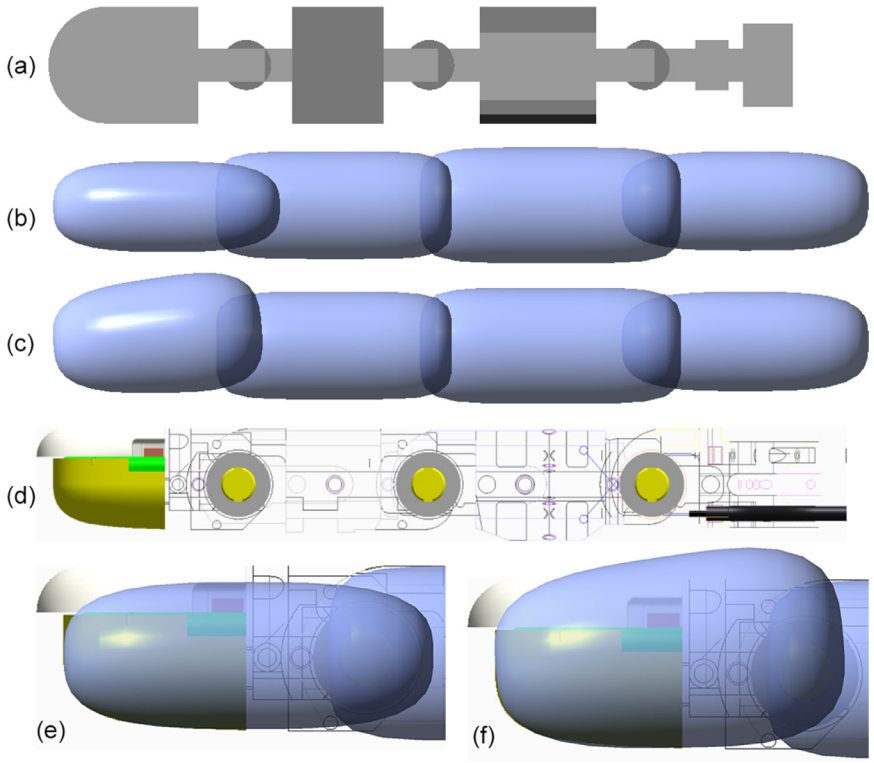

Fig. 7. Approximation of the iCub's right index finger using different representations. (a) simplified sphere and cylinder meshes from [37]; (b) proposed superellipsoid representation; (c) proposed superovoid representation; (d) original CAD illustration from iCub repository; (e) close-up of the fingertip in (b) superimposed on the CAD image; (f) close-up of the fingertip in (c) superimposed on the CAD image.

Because FCL's mesh collision implementation uses a bounding volume hierarchy (BVH) strategy (specifically OBB and Rectangle Swept Sphere volumes), and not the raw vertices and triangles, its broad-phase and narrow-phase parts could not be separated for testing. Thus, the superovoid and superellipsoid collision queries were executed only after a narrow-phase culling consisting of a bounding sphere test, followed by an OBB test. For a meaningful comparison, the algorithms were only timed when the queries reported a collision, that is, when the objects were in fact intersecting.

5.3. Dental occlusion

In order to provide a fairer comparison between analytical and mesh-based contact geometries, a more complex object than the iCub fingers was tested. The added complexity consists of a conformal situation, where a set of superovoids are arranged to fit the non-convex surfaces of the dental crowns of two molar teeth. Both structures exhibit intricate surfaces geometries with several depressions and elevations. Therefore, multiple points of contact appear during bite as each elevation is approximated by a superovoid contact geometry (Fig. 8).

The mesh was reconstructed after segmenting CT data of a mandible using an appropriate image-based anatomical modeling framework [38,41]. The dental scan of the entire lower jaw (CT Siemens Sensation 64, 512×512 acquisition matrix, in-plane x and y resolutions 0.36×0.36 mm^2, slice thickness 0.75 mm, and 166 slices) is available from the OsiriX DICOM image sample set. Consent for the use of the CT data set was provided by the subject. For each tooth, superovoids were manually adjusted to the segmented triangular mesh to represent individual elevations. This collection of superovoids was then polygonized using the marching triangles algorithm to obtain a mesh representation of the object modeled with multiple superovoidal meshes.

A prescribed linear motion was defined to emulate the bite between the upper jaw tooth that descends towards the fixed lower jaw tooth. To ensure rapid computations, a hierarchy of AABBs was considered as the acceleration structure for the meshless approach, i.e., each superovoid was bounded by an axis aligned box. Whenever the proximity query was true, the corresponding Newton-Raphson method was then executed to compute the minimum distance points between contact surface pairs. The initial iteration for the Newton Raphson algorithm was obtained with the parametric quadtree method, hence, for this case study the quadImplicit scheme was considered and compared to mesh geometries with different triangular resolutions.

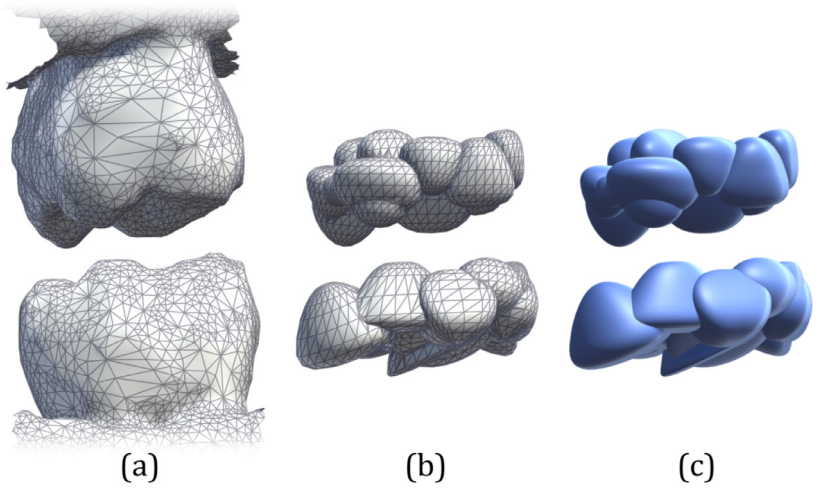

Fig. 8. Molar teeth representations approximated by (a) segmented triangular meshes, (b) mesh of a set of superovoids, and (c) set of analytical superovoids.

Table 1.
Comparison between a pair of overlapping superovoids with quadImplicit, quadParametric and ntbParametric Newton-Raphson (N-R) algorithms, when starting from the same initial iteration, as well as the FCL mesh-based minimum distance algorithm for a mesh with 290 vertices. Time is expressed in micro seconds (μs). Total time includes initial guess time. ($\mu \pm \sigma$: average ± standard deviation).

Representation	Initial guess time	Total time	N-R iterations	Time per N-R iteration
quadImplicit	129.28 ± 9.26	310.42 ± 39.50	6.20 ± 1.23	21.20
quadParametric	129.28 ± 9.26	424.53 ± 106.50	6.93 ± 2.56	31.59
ntbParametric	129.28 ± 9.26	347.72 ± 78.31	6.92 ± 2.52	22.99
Mesh (FCL)	–	206.29 ± 117.63	–	–

6. Results and discussion

A benchmark study consisting of five trials was performed to compare different contact geometry representations (mesh triangulation, implicit and parametric surfaces) regarding computational performance, geometric accuracy and applicability to real world mechanisms. Note that benchmarking the proposed algorithms with other methods found in the literature is out of the scope of our paper as our focus is the influence of contact geometry representation on closest distance calculation between superovoids and superellipsoids in a non-conformal configuration, during the narrow-phase of contact either apart or slightly overlapping.

The results refer mainly to the distance computation between contact pairs of superellipsoidal and superovoidal shapes in different spatial configurations with varying surface parameters. Even though the computation of the minimum distance between smooth convex objects in a non-conformal configuration provides a quantitative figure on the collision state, the proposed approach cannot be considered as a proximity query as more efficient methods exist to determine if two surfaces are apart (e.g., AABB, OBB, Bounding Spheres), yet, such methods only output the qualitative contact state and not a numerical distance between surfaces. No multibody dynamic calculations are undertaken in order to evaluate, solely, the distance computation efficiency during the analysis. This section is organized according to the case studies under evaluation.

6.1. Separated and overlapping contact pairs

The following trial was performed involving 10.000 random pairs of surfaces (as described in Section 5.1), and its results are shown in Tables 1 and 2 which summarize the major differences observed experimentally when using Eq. (15) instead of Eq. (18).

Providing the quadImplicit, quadParametric and ntbParametric versions of the algorithm with the same initial iteration for the Newton-Raphson procedure allows a more insightful comparison of the three methods. The quadImplicit version is faster than the other two parametric representations, which means that the implicit formulas for computing normals, tangents, bitangents and the subsequent solution of the system of 6 non-linear equations is faster than its parametric 4×4 counterpart. In part, this is due to the implicit version needing less iterations to converge, but it should be noted that the

Table 2.
Comparison between a pair of separated superovoids with quadImplicit, quadParametric and ntbParametric Newton-Raphson (N-R) algorithms, when starting from the same initial iteration, as well as the FCL mesh-based minimum distance algorithm for a mesh with 290 vertices. Time is expressed in micro seconds (μs). Total time includes initial guess time. ($\mu \pm \sigma$: average ± standard deviation).

Representation	Initial guess time	Total time	N-R iterations	Time per N-R iteration
quadImplicit	129.28 ± 9.26	290.55 ± 36.35	5.47 ± 1.10	21.26
quadParametric	129.28 ± 9.26	396.00 ± 107.77	6.20 ± 2.55	31.72
ntbParametric	129.28 ± 9.26	326.41 ± 79.99	6.20 ± 2.54	23.05
Mesh (FCL)	–	375.19 ± 104.74	–	–

Table 3.
Qualitative comparison of pros and cons of the proposed and studied geometries and algorithms, in terms of computation time (average and standard deviation), required memory, and geometric accuracy. ($\mu \pm \sigma$: average ± standard deviation).

Representation	Time (μ)	Time (σ)	Memory	Accuracy
Meshes (BVH collision detection)	fastest	low	high	low
Meshes (minimum distance)	slow	low	high	low
Superellipsoids/superovoids implicit	fast	high	low	high
Superellipsoids/superovoids parametric	faster	high	low	high

computation time of one single implicit iteration is also lower. The ntbParametric version of the algorithm employs the implicit formulas to compute the normals, tangents and bitangents, which results in a quicker processing when compared to the purely parametric method, but the implicit version was still faster on average. The quadImplicit version is also faster than the mesh-based algorithm in FCL for the case of separated superovoids (Table 2), even when used with relatively simple (and geometrically imprecise) models of 290 vertices. In general, the studied algorithms converged in less iterations for the separated case (Table 2) than for the overlapping case (Table 1). This is due to the fact that the initial iteration provided by the parametric quadtree algorithm (see Section 4.4) converges to the intersection between the two superovoids if they are overlapping, instead of the minimum distance points given by the common normal concept.

Finally, it is worth noting that, using either the implicit or parametric representations, it is possible to fully describe a superovoid storing only 7 (shape) + 6 (rigid body) parameters stored as floating point numbers, while polygonal meshes occupy considerably more memory to store vertices, triangles and eventual BVH geometries. Thus, the pros and cons of using implicit or parametric superellipsoids or superovoids for minimum distance computations, compared to meshes, are qualitatively summarized in Table 3.

Due to the local nature of the Newton-Raphson method, performance is conditioned mainly by how close is the initial approximation towards the desired solution. Therefore, placing initial approximations in different areas of the surface should converge slower or faster according to how well the common normal conditions are satisfied. To better guarantee a more controlled step setting, one could consider the Baumgarte stabilization approach.

Another aspect that can influence performance is due to the non-linear nature of superovoids that expresses a wide range of curvature values. In fact, surface curvature can influence the iteration step that the following iteration must take as tight curved areas require a smaller step whereas loose or more planar curved areas require a larger step.

Our approach relies on a quadtree subdivision that adapts according to the minimum distance between surfaces and common normal conditions, and not to the curvature values. Note that curvature information is unrelated to the common normal concept.

6.2. Geometric accuracy

An isolated test of the geometric accuracy of the algorithms was run, using 2 superovoids and comparing them to the 2 corresponding meshes generated from discretizations with varying resolution. Its results can be seen in Fig. 9 and indicate that both implicit and parametric contact geometry representations of superovoids are efficient and accuracy-controlled.

Regarding the geometric precision of the superovoid and superellipsoid queries, these can be easily adjusted, while for mesh-based algorithms, this would involve regenerating the mesh with a different number of vertices and triangles, which might not be feasible in real-time. The reported contact normal for mesh-based algorithms can also be noisy due to the faceted nature of polygonal meshes and strictly depends on its tessellation, which does not happen for the proposed algorithms. As shown in Fig. 9, the mesh-based algorithm only achieves a precision of 10^{-1} m using more than 786 vertices (see Fig. 10); the superovoid version's precision is 10^{-6} m and takes an order of magnitude less time to compute. Furthermore, the superovoid query is faster than the mesh one, except for meshes with 18 vertices or less, and these meshes lead to very low geometric precision.

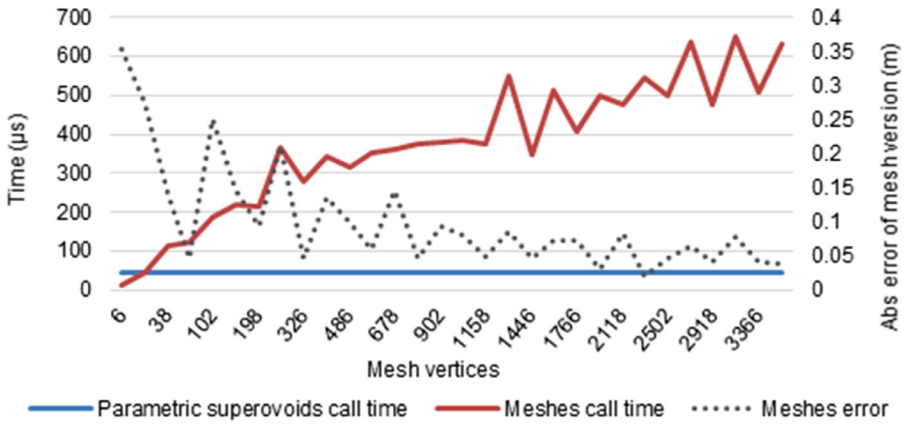

Fig. 9. Comparison of computation times and geometric accuracy between the proposed parametric superovoid algorithm and the FCL mesh minimum distance query for a single contact pair of superovoids. The meshes are in the shapes of superovoids, of length parameters a_1, a_2, $a_3 \in [0.96, 1.4]$m, with different resolutions.

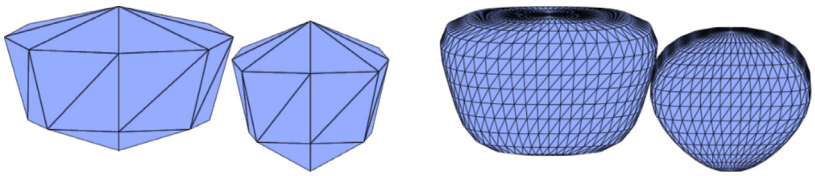

Fig. 10. Meshes of superovoids used for the accuracy benchmark of Fig. 9, in their relative positions. On the left, meshes with 18 vertices, 32 triangles. On the right, meshes with 786 vertices, 1568 triangles.

Table 4.
Minimum distance computation times (average and standard deviation) and number of Newton-Raphson iterations (average), taken from around 1600 queries, for FCL's mesh implementation and for the proposed algorithms for the iCub Hands case. Times correspond to the full minimum distance computation process.

Representation	Average call time (µs)	Standard deviation of call time (µs)	Average Newton-Raphson iterations
Meshes (BVH proximity query)	28.29	21.98	–
Meshes (minimum distance)	181.03	95.21	–
Superellipsoids (implicit)	127.71	299.89	3.7
Superovoids (implicit)	123.55	309.49	3.5
Superellipsoids/superovoids (implicit)	128.62	277.83	2.7
Superellipsoids (parametric)	39.01	96.27	2.3
Superovoids (parametric)	45.67	95.11	2.1
Superellipsoids/superovoids (parametric)	49.55	116.65	2.3

6.3. iCub Hand representation

The following trial evaluates the contact behavior of the iCub Hand (as described in Section 5.1). A total of 8 tests were performed in MoveIt! with ROS Indigo, both compiled from source, on a Lubuntu 32 bit system with an Intel® Core™ i7-4700MQ @ 2.40 GHz and 4 GB of RAM. Each test corresponds to a different contact geometry representation, shape or initial iteration technique. The time results of these tests are shown in Table 4 and Fig. 11.

The results in Table 4 indicate that the proposed algorithms run 1.5 to 4 times faster than their mesh equivalent. As for FCL's mesh collision detection query, despite being significantly faster than the minimum distance queries, it uses an OBB hierarchy to accelerate the computation at the expense of geometric accuracy, hence it cannot be considered a minimum distance technique but rather a proximity query method.

The algorithm for implicit surfaces is considerably slower than the parametric version, due to the octree-based initial estimation being quite expensive (500 µs per estimation, on average), and a new estimation must be calculated if the Newton-Raphson algorithm does not converge using the result from the previous time-step as the initial iteration. The parametric quadtree-based estimation method is simpler, as it is defined in a 2D domain, and the parametric geometric constraints, while requiring more non-linear functions evaluations due to the co-sines and sines, guarantee that the potential points are on the surfaces and the numerical method diverges less often. Still, these spurious needs for new estimations increase the time standard deviation of the proposed algorithms, especially for the implicit versions. As shown in Section 6.1, the

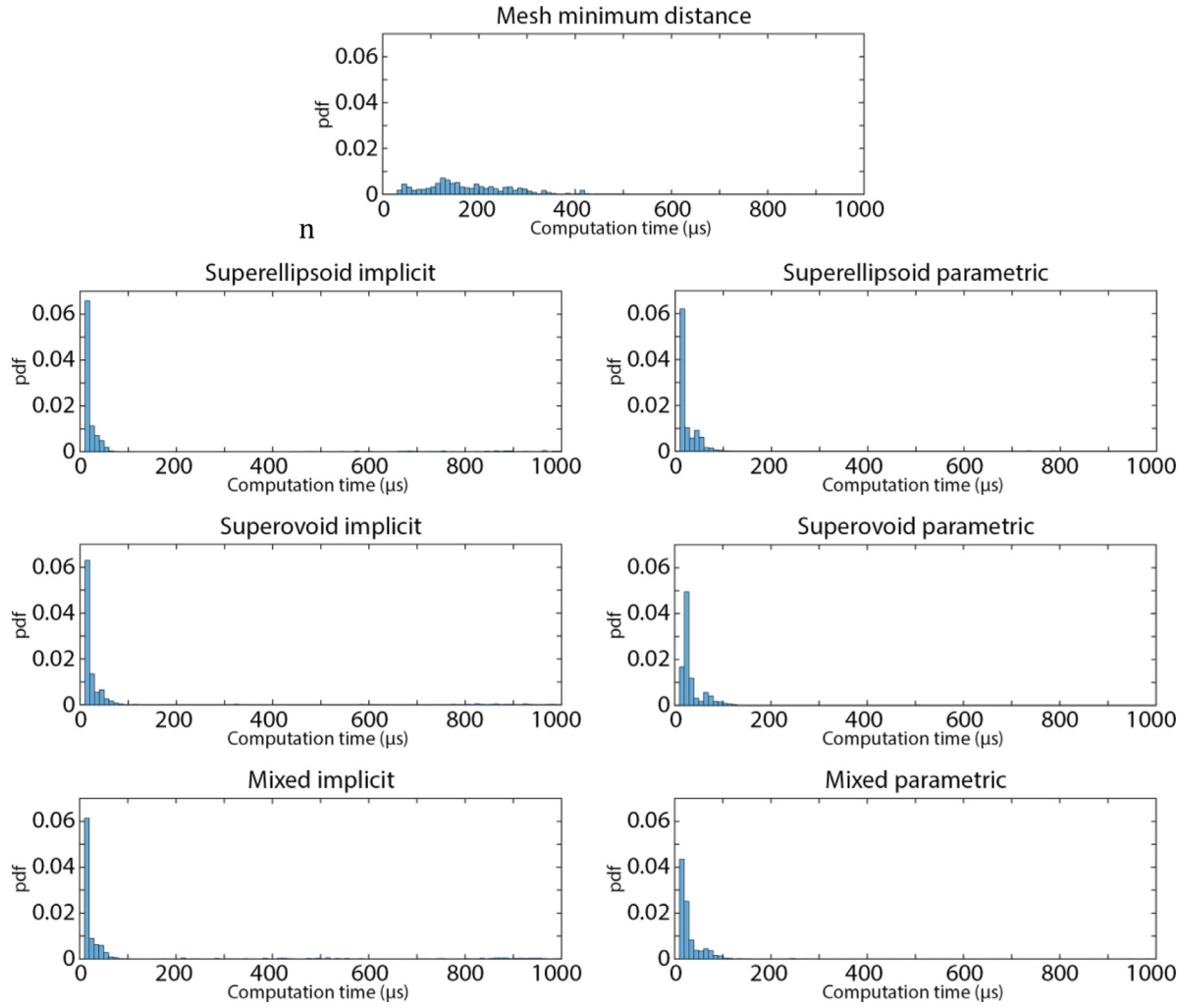

Fig. 11. Computation time distributions of the minimum distance queries, comparing FCL's mesh implementation with the proposed methods for the iCub Hands case. In all cases the average and median times are lower compared to the mesh algorithm. In some rare cases, the proposed methods can have high computation times (1 ms) if the Newton-Raphson procedure diverges and has to start over from a new initial estimation (more noticeable in the implicit representations). pdf: probability density function.

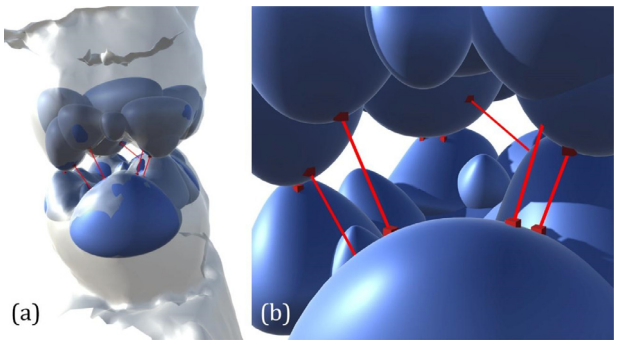

Fig. 12. Superovoid representation of two molar teeth in a bite motion. In (a), the original teeth mesh is shown superimposed in grey so the position of the teeth is easier to understand – this mesh was not used in simulations. In (b), the minimum distances between several pairs of superovoids is shown in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

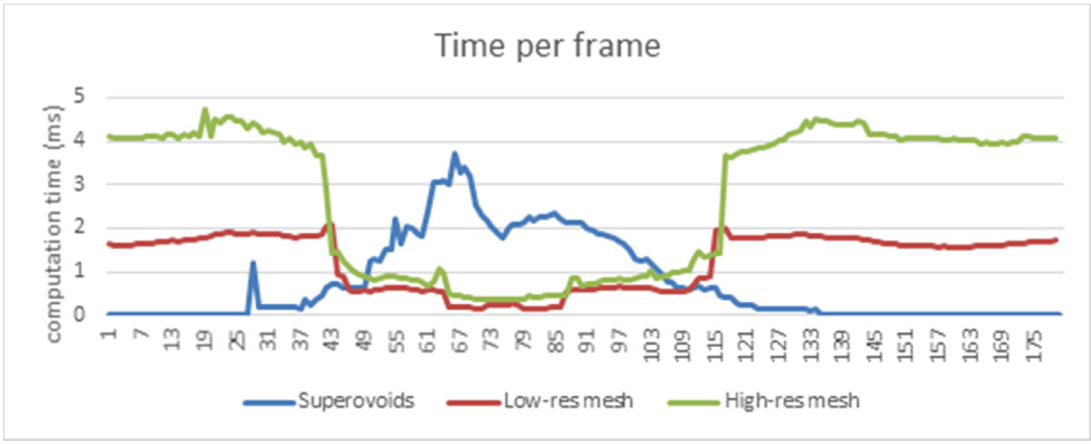

Fig. 13. Computation time per frame of animation, for each algorithm. The teeth intersect around frame 47 and separate around frame 111.

implicit version runs faster than the parametric one when both are given the same initial iteration, computed with the same method.

The simpler superellipsoid versions were not significantly different than the general superovoid ones in terms of computation time, even though the mathematical expressions for computing normals and values of the implicit function were slightly faster when tested in isolation. Although superovoids share a similar computational performance with superellipsoids, this can be seen as an advantage as there is no relevant additional cost when considering a more flexible and non-linear shape that is capable to represent a greater number of objects.

6.4. Dental occlusion

We compared mesh based approaches with superovoid approaches for two molar teeth during bite. Multiple superovoids were used to represent the teeth crowns. After plotting the minimum distance vs time, we get a quantitative measure of the tradeoff between accuracy (measured as the difference between minimum distances computed for each contact geometry) and decrement of computational complexity. These results show that quadImplicit superovoid performs better than meshes.

The simulated bite motion consists of 181 frames of animation, in which the upper tooth moves towards the lower tooth, and then rises again. The following algorithms were compared: (i) superovoid representation with implicit Newton Raphson and parametric initial iteration, (ii) FCL mesh algorithm with 6200 triangles per tooth, (iii) FCL mesh algorithm with 24,000 triangles per tooth.

The algorithms have different behaviors depending on whether the teeth are overlapping or not (see Fig. 13). The mesh-based minimum distance algorithm from FCL is faster when the teeth are overlapping, while the superovoid-based algorithm is faster when they are separated. For the case of the superovoids, this is because the AABBs for many superovoids are not overlapping, and thus the Newton Raphson algorithm is never called, and the AABB check is computationally inexpensive. As more pairs of superovoids come into proximity of each other, the superovoid-based algorithm takes more time to compute. Note, however, that the mesh-based algorithm does not report penetration distance when the teeth are in contact – it only returns a distance of zero (see Fig. 14), while the superovoid algorithm returns the penetration distance for each pair of overlapping superovoids.

7. Conclusion

Estimating the closest distance for contact simulations demands increasingly higher geometric accuracy without trading off computational efficiency or adversely affect valuable computational resources. As far as collision detection with smooth convex surfaces is concerned, polygonal meshes and bounding volume hierarchies are considered the gold standards both in literature and industry, but they both entail difficulties: meshes must be very detailed to achieve high geometric precision, and this slows down mesh-based algorithms, while BVH are by design, implemented for performing very fast, low-accuracy proximity queries.

This work analyzed smooth convex surfaces and their implicit and parametric representations for efficient real-time minimum distance calculation between surfaces in a non-conformal configuration, which still is a very challenging topic. In particular, we studied both the superellipsoid and superovoid geometries, and applied them to various case-studies, including single contact pairs (i.e., two interacting surfaces) or combinations of contact pairs that interact in either non-conformal (e.g., iCub fingers) or conformal (e.g., dental occlusion) fashion. Our experiments demonstrate that implicit contact geometry representations are adequate to implement efficient and precision-controlled algorithms to compute the minimum distance between superovoidal contact surfaces. Results indicate that parametric and implicit representations are more computationally efficient than standard mesh-based contact geometries.

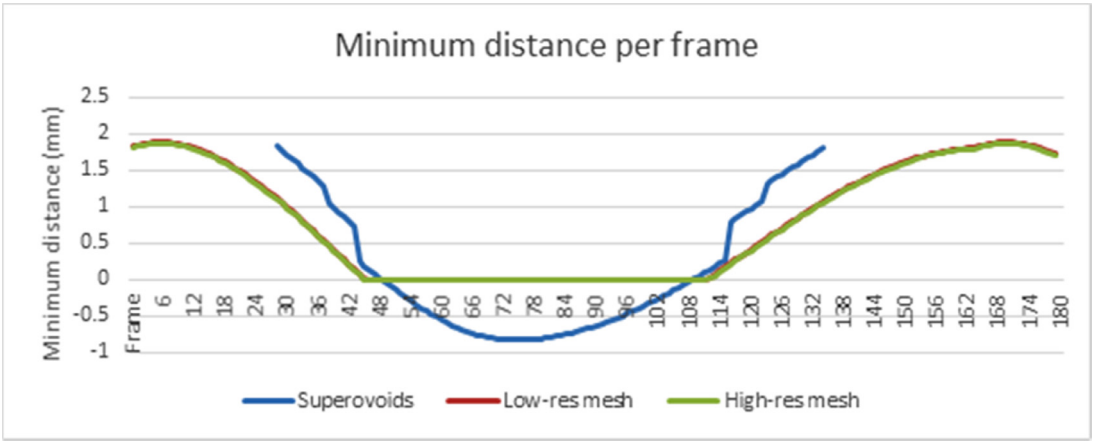

Fig. 14. Minimum distance between the teeth on each frame of animation. Note that the FCL mesh minimum distance algorithm does not report penetration distance, so the mesh results between frames 45 and 111 are zero. The negative values for the superovoid algorithm during these frames are penetration distances. The discontinuities in the superovoid version happen when the AABBs of a pair of superovoids begin to intersect, and the Newton Raphson algorithm starts reporting minimum distances. Before frame 28 and after frame 133, there is no Superovoid data, because there are no AABBs intersecting.

We demonstrated that superovoid and superellipsoid models can be applied to calculate closest distances for robotic grasp planning, by implementing the analytical contact geometries in FCL and using them together with MoveIt! to simulate the iCub's hand. Both implicit and parametric representations are compared, and the algorithms both perform faster than FCL's mesh-based minimum distance implementation, with the added advantages of representing the contact geometry with less memory compared to meshes, while achieving arbitrarily accurate contact points and normals. To the authors' knowledge, the (super)ovoid shape has never been used in robotic grasping simulation. Considering conformal contact situations between a pair of superovoids will also be considered in the future. In fact, if one or both surfaces are concave, multiple solutions may appear for Eqs. (15) and (18). Therefore, at each time step it is necessary to check whether the obtained solution matches the minimum distance.

There are several improvements and future avenues worth looking into. An accurate starting estimate is critical to make the Newton-Raphson algorithm converge to the correct solution. Methods to obtain this starting iteration should be further compared, and new, faster and more precise methods can be investigated. Since real-time computation is a key desideratum of our work, we opted for a simple partition method that recursively subdivides the angular domain into four quadrants or regions. Note that the quadtree method can be considered as a 2-D equivalent of the bisection technique that converges globally to the solution which lies inside some interval. Also take in account that a quadtree enumeration of the space adapts to the curvature by further subdividing the surface in areas of higher curvature.

Despite comparing different initial approximation strategies for a given surface representation being out of the scope of our work, this may be an interesting direction for future work. In fact, according to [42], a non-uniform and/or adaptive partition scheme could improve performance as compared to the homogeneous quadtree method. This makes angular domain discretization an interesting future topic for improving initial approximation estimation.

The Jacobian matrix used in the numerical procedure can be analytically computed, which should lead to shorter execution times. In addition, the proposed algorithm should be compared to other minimum distance approaches such as those proposed by [38–40] that adopt a non-linear optimization approach, and [5] which considers a particular parametric formulation for superellipsoids. The comparison criteria would rely on computational efficiency, accuracy and robustness.

Smooth convex surfaces, in particular superovoids, may also provide adequate representations for other parts of robots, such as the iCub's arms and torso. In order to improve the accuracy of the robot finger geometric model, an automated surface fitting approach should be considered. Besides robotics, the proposed algorithms and contact geometries should be employed and tested in other contact scenarios that are approachable to experimental validation. We conclude that implicit formulations are both more compact, simpler to evaluate and allow better computational resource tradeoffs in critical simulations, where precise calculations of contact force and torque are paramount to achieving quality simulations.

Acknowledgements

All authors are thankful for the financial support given by Portuguese Foundation for Science and Technology (FCT). In particular, the second author thanks for the postdoctoral grant SFRH/BPD/97449/2013. This work was also partially supported by national funds through FCT through projects UID/EEA/50009/2013, UID/CEC/50021/2013 and IT-MEDEX PTDC/EEI-SII/6038/2014.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.mechmachtheory.2017.04.008.

References

[1] F. Pfeiffer, C. Glocker, Multibody Dynamics With Unilateral Constraints, John Wiley and Sons, New York, 1996.
[2] K. Johnson, Contact Mechanics, Cambridge University Press, 1985.
[3] G. van den Bergen, Efficient collision detection of complex deformable models using AABB trees, J. Graph. Tools 2 (4) (1997) 1–13.
[4] A. Barr, Superquadrics and Angle-Preserving Transformations, Comput. Graph. Appl. IEEE 1 (1) (1981) 11–23.
[5] C. Wellmann, C. Lillie, P. Wriggers, A contact detection algorithm for superellipsoids based on the common-normal concept, Eng. Comput. 25 (5) (2008) 432–442.
[6] D. Lopes, M. Silva, J. Ambrósio, P. Flores, A mathematical framework for contact detection between quadric and superquadric surfaces, Multibody. Sys. Dyn. 24 (3) (2010) 255–280.
[7] K. Moustakas, D. Tzovaras, M.G. Strintzis, SQ-Map: Efficient Layered Collision Detection and Haptic Rendering, IEEE Trans. Vis. Comput. Graph. 13 (January(1)) (2007) 80–93.
[8] N. Chakraborty, S. Berard, S. Akella e, J.C. Trinkle, Proximity queries between convex objects: An interior point approach for implicit surfaces, Robot. IEEE Trans. 24 (2008) 211–220.
[9] D.S. Lopes, R.R. Neptune, A.A. Gonçalves, J.A. Ambrósio, M.T. Silva, Shape analysis of the femoral head: a comparative study between spherical, (super)ellipsoidal, and (super)ovoidal shapes, J. Biomech. Eng. 137 (October(11)) (2015) 114504–114504-8.
[10] P. Todd, I. Smart, The shape of birds' eggs, J. Theor. Biol. 106 (2) (1984) 239–243.
[11] S. Chen, S. Xin, Y. He, G. Wang, The Closest and Farthest Points to an Affine Ellipse or Ellipsoid, Tsinghua Sci. Technol. 17 (August(4)) (2012) 481–484.
[12] A. Pazouki, H. Mazhar e, D. Negrut, Parallel collision detection of ellipsoids with applications in large scale multibody dynamics, Math. Comput. Simul 82 (5) (2012) 879–894.
[13] X. Jia, Y.-K. Choi, B. Mourrain, W. Wang, An algebraic approach to continuous collision detection for ellipsoids, Comput. Aided Geometric Design 28 (3) (2011) 164–176.
[14] I. Gilitschenski, U.D. Hanebeck, a robust computational test for overlap of two arbitrary-dimensional ellipsoids in fault-detection of Kalman filters, *Information Fusion (FUSION), 2012 15th International Conference on*, Singapore, 2012.
[15] K. Moustakas, G. Nikolakis, D. Koutsonanos, D. Tzovaras, M. Strintzis, Haptic feedback using an efficient superquadric based collision detection method, Eurohaptics Conference, 2005 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, 2005. World Haptics 2005. First Joint, 2005.
[16] G. Lu, J. Third, C. Müller, "Critical assessment of two approaches for evaluating contacts between super-quadric shaped particles in DEM simulations, Chem. Eng. Sci. 78 (2012) 226–235.
[17] G. Lu, J. Third, C. Müller, Discrete element models for non-spherical particle systems: From theoretical developments to applications, Chem. Eng. Sci. 127 (2015) 425–465.
[18] D. Lopes, R. Neptune, J. Ambrósio, M. Silva, A superellipsoid-plane model for simulating foot-ground contact during human gait, Comput. Methods Biomech. Biomed. Eng. (September) (2015) 1–10.
[19] D. Lopes, M. Silva, J. Ambrósio, Tangent vectors to a 3-D surface normal: A geometric tool to find orthogonal vectors based on the Householder transformation, Comput. Aided Des. 45 (3) (2013) 683–694.
[20] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. v. Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, A. Bernardino, L. Montesano, The iCub humanoid robot: an open-systems platform for research in cognitive development, Neural Netw. 23 (8-9) (2010) 1125–1134.
[21] R.M. Murray, S.S. Sastry, L. Zexiang, A Mathematical Introduction to Robotic Manipulation, in: FL Boca Raton (Ed.), CRC Press, Inc, USA, 1994.
[22] Z. Xue, P. Woerner, J. Zoellner, R. Dillmann, Efficient grasp planning using continuous collision detection, Mechatronics and Automation, 2009. ICMA 2009. International Conference on, 2009.
[23] J. Shi, G.S. Koonjul, Real-time grasp planning with environment constraints, *IROS 2014 Workshop: Real-time Motion Generation & Control*, Chicago, 2014.
[24] T. Tsuji, S. Uto, K. Harada, R. Kurazume, T. Hasegawa, K. Morooka, Grasp planning for constricted parts of objects approximated with quadric surfaces, *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, Chicago, IL, 2014.
[25] C. Qian, X. Sun, Y. Wei, X. Tang e, J. Sun, Realtime and robust hand tracking from depth, em *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
[26] I. Oikonomidis, N. Kyriazis, A. Argyros, Efficient model-based 3D tracking of hand articulations using Kinect, in: Proceedings of the British Machine Vision Conference, 2011.
[27] M.A. Sherman, A. Seth, S.L. Delp, Simbody: multibody dynamics for biomedical research, Procedia IUTAM (2) (2011) 241–261.
[28] N. Koenig, A. Howard, Design and use paradigms for gazebo, an open-source multi-robot simulator, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004.
[29] H. Taeubig, U. Frese, A new library for real-time continuous collision detection, in: *Robotics; Proceedings of ROBOTIK 2012; 7th German Conference on*, Munich, 2012.
[30] J. Pan, S. Chitta, D. Manocha, FCL: a general purpose library for collision and proximity queries, *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, Saint Paul, MN, 2012.
[31] I.A. Sucan, S. Chitta, MoveIt!, 2015 [Online]. Available: http://moveit.ros.org. [Accessed 30 08 2015].
[32] G. Farin, Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide, fifth ed, Academic Press, San Diego, CA, 2002.
[33] M. Duncan, Applied Geometry for Computer Graphics and CAD, second ed., Springer, 2005.
[34] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, Octomap: an efficient probabilistic 3d mapping framework based on octrees, Autonom. Robots 34 (3) (2013) 189–206.
[35] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D Sorensen, LAPACK Users' Guide, third ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1999.
[36] M. Quigley, K. Conley, B.P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A.Y. Ng, ROS: an open-source robot operating system, ICRA Workshop on Open Source Software, 2009.
[37] VisLab, ISR Lisboa, iCub MoveIt! URDF model, 2014 [Online]. Available: https://github.com/vislab-tecnico-lisboa/icub-moveit. [Accessed 30 August 2015].
[38] "OsiriX DICOM image library," [Online]. Available: http://www.osirix-viewer.com/resources/dicom-image-library/.
[39] N. Chakraborty, J. Peng, S. Akella, J. Mitchell, Proximity queries between convex objects: an interior point approach for implicit surfaces, Robotics, IEEE Transactions on, 2008.
[40] M. Hopkins, Discrete element modeling with dilated particles, Eng. Comput. 21 (2/3/4) (2004) 422–430.
[41] N.S. Ribeiro, P.C.R. Fernandes, D.S. Lopes, J. Folgado, P.R.A. Fernandes, Solid and finite element modeling of biomechanical structures: a software pipeline, *7th EUROMECH Solid Mechanics Conference*, Lisbon, Portugal, 2009.
[42] R. Pajarola, M. Antonijuan, R. Lario, QuadTIN: quadtree based triangulated irregular networks, in: Proceedings of the conference on Visualization '02 (VIS '02), Washington, DC, USA, IEEE Computer Society, 2002, pp. 395–402.