Fast oriented bounding box optimization on the rotation group $\boldsymbol{SO(3,\mathbb{R})}$

CHIA-TCHE CHANG[1], BASTIEN GORISSEN[2,3] and SAMUEL MELCHIOR[1,2]
[1]Université catholique de Louvain, Department of Mathematical Engineering (INMA)
[2]Université catholique de Louvain, Applied Mechanics Division (MEMA)
[3]Cenaero

An exact algorithm to compute an optimal 3D oriented bounding box was published in 1985 by Joseph O'Rourke, but it is slow and extremely hard to implement. In this article we propose a new approach, where the computation of the minimal-volume OBB is formulated as an unconstrained optimization problem on the rotation group $SO(3,\mathbb{R})$. It is solved using a hybrid method combining the genetic and Nelder-Mead algorithms. This method is analyzed and then compared to the current state-of-the-art techniques. It is shown to be either faster or more reliable for any accuracy.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Geometric algorithms*; *Object representations*; G.1.6 [**Numerical Analysis**]: Optimization—*Global optimization*; *Unconstrained optimization*

General Terms: Algorithms, Performance, Experimentation

Additional Key Words and Phrases: Computational geometry, optimization, manifolds, bounding box

1. INTRODUCTION

This article deals with the problem of finding the minimum-volume *oriented bounding box* (OBB) of a given finite set of N points, denoted by $\mathcal{X} \subset \mathbb{R}^3$. The problem consists in finding the cuboid, i.e., rectangular parallelepiped, of minimal volume enclosing $\mathcal{X}$. This is illustrated for the 2D case in Fig. 1. Each OBB is defined by its center $\mathbf{X} \in \mathbb{R}^3$, its dimension $\boldsymbol{\Delta} \in \mathbb{R}^3$ and its orientation $R \in SO(3,\mathbb{R})$. This rotation group is the special orthogonal group of degree 3 over $\mathbb{R}$:

$$SO(3,\mathbb{R}) = \left\{R \in GL(3,\mathbb{R}) \mid R^T R = I = RR^T, \det(R) = 1\right\},$$

where $GL(3,\mathbb{R})$ is the general linear group of degree 3, i.e., the set of 3-by-3 invertible real matrices. The matrix R rotates the reference frame $\mathbf{e}_x$ onto $\mathbf{e}_\xi$ as shown in Fig. 1. The convex hull of $\mathcal{X}$ and the set of its vertices are denoted by $\mathcal{CH}(\mathcal{X})$ and $\mathcal{X}^C \subset \mathcal{X}$, respectively. Just as $N = |\mathcal{X}|$, let $N_V = |\mathcal{X}^C|$ be the number of vertices of $\mathcal{CH}(\mathcal{X})$. This convex hull can be obtained as a preprocessing step with cost $\mathcal{O}(N \log N)$.

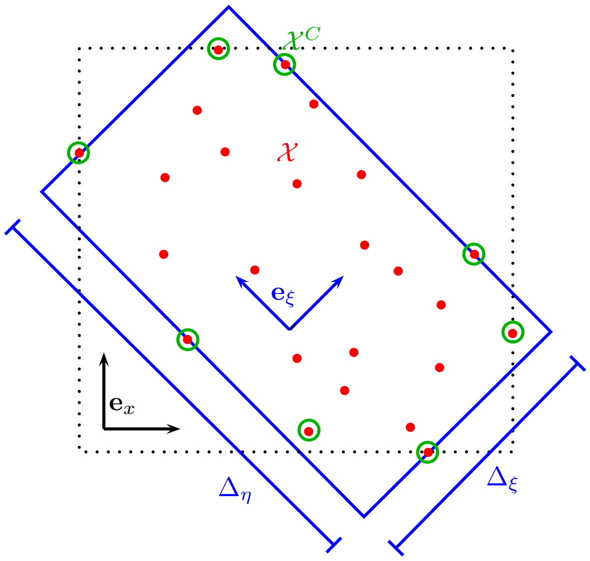


Fig. 1. Illustration of some of the notation used throughout the article. The axis-aligned bounding box and the optimal oriented bounding box are drawn as dotted black and solid blue lines, respectively.

Computing the minimal OBB is far from trivial, although it can be solved in polynomial time using O'Rourke's exact algorithm [O'Rourke 1985]. Approximations can also be obtained using heuristics such as the ones based on *principal component analysis* (PCA) [Jolliffe 2002]. In this article, the orientation of an optimal OBB is computed by a hybrid global optimization algorithm that searches in the space of rotation matrices. Optimal OBBs of about 300 test cases can be estimated with relative accuracy of 1 % or better in 98 % of the runs on average. Running the algorithm once

This article presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with the authors.
Authors' addresses : *Chia-Tche Chang* (corresponding author), Department of Mathematical Engineering, Université catholique de Louvain, 4, avenue Georges Lemaître, 1348 Louvain-la-Neuve, Belgium, email: `chia-tche.chang@uclouvain.be`; *Bastien Gorissen*, Cenaero, 29, rue des Frères Wright, 6041 Gosselies, Belgium; *Samuel Melchior*, Applied Mechanics Division, Université catholique de Louvain, 4, avenue Georges Lemaître, 1348 Louvain-la-Neuve, Belgium.

DOI 10.1145/2019627.2019641
http://doi.acm.org/10.1145/2019627.2019641

on each test case takes about 4 minutes. In comparison, PCA-based methods only reach such accuracy in less than 60 % of the cases. For another choice of parameters of our method, we were able to compute all optimal volumes with a relative error of at most 10^{-12} in more than 95 % of the runs on average, in about 20 minutes. In comparison, O'Rourke's method requires nearly one week of computation time to compute one optimal OBB for each example. All these computation times include the computation of the convex hulls.

The OBB fitting problem is encountered in many applications, such as fast rendering of 3D scenes [Assarsson and Möller 2000], collision detection [Jiménez et al. 2000; Garcia-Alonso et al. 1994], visibility tests [Iones et al. 1998] or mesh reparameterization [Remacle et al. 2010]. Those applications use a simplified hierarchical representation of 3D objects (see [Gottschalk et al. 1996; Ponamgi et al. 1997]). The various 3D models are usually approximated by a tree of successively smaller volumes ([Gottschalk et al. 1996] uses such a method), requiring two characteristics:

(1) The volumes themselves, and an intersection test between two such volumes must be quickly computable.
(2) Those volumes need to be as close as possible to the geometry defined by $\mathcal{X}$ in order to minimize the number of superfluous tests and thus improve the total running time of the algorithms.

On one side of the spectrum lies the *axis-aligned bounding box* (AABB, see [den Bergen 1998]), i.e., the OBB with R chosen as the identity matrix I. This is the easiest bounding volume to compute, as only the range of the coordinates of the points in $\mathcal{X}$ needs to be evaluated. However, it is obviously not very good at fitting most geometries.

On the opposite side, the convex hull of the model provides by definition the closest convex approximation, but collisions between these convex polyhedra can be just as hard to detect as those involving the original models. This is why arbitrary OBBs are often a good compromise.

Other bounding volumes can of course be used, such as bounding spheres, bounding cylinders, bounding capsules (cylinders capped by two half-spheres) or k-DOPs (AABB with beveled edges and corners), all of which compromise computing ease and accuracy differently (a review of various bounding volumes can be found in [Ericson 2004]).

The main idea behind this article is that the problem of finding an optimal OBB can be written as an unconstrained optimization problem on $SO(3, \mathbb{R})$. Hence, the objective function is continuous but non-differentiable as it takes into account the geometric constraints. Therefore, it cannot be written in closed form although it is easy to evaluate. This is why solving such a formulation of the problem requires the use of derivative-free optimization methods, such as those used in the Hybrid Bounding Box Rotation Identification (HYBBRID) algorithm we propose. It consists in a hybridization of the genetic and Nelder-Mead algorithms, based on the method described in [Durand and Alliot 1999]. Such a hybrid scheme combines the strength of the genetic algorithm in terms of exploration of the search space, and the capacity of the Nelder-Mead method to quickly converge to locally good solutions.

The article is organized as follows. It starts with an extensive review of the literature on the subject. Next, the formulation of the OBB fitting problem as an optimization problem is detailed. Our HYBBRID algorithm is then described, analyzed and compared to the other methods.

2. STATE OF THE ART

The problem of computing an optimal OBB for a given set of points is not trivial. In 2D, an optimal bounding rectangle can be computed in linear time using the so-called *rotating calipers* method as proposed in [Toussaint 1983]. This technique is based on the idea developed by Michael Ian Shamos in his Ph.D. thesis [Shamos 1978] (see also [Preparata and Shamos 1985]) to compute the diameter of a convex polygon. The current best exact algorithm for the 3D problem, published by O'Rourke in 1985 [O'Rourke 1985], has a time complexity of $\mathcal{O}(N_V^3)$. O'Rourke's algorithm is too slow to be of practical use and is known to be extremely hard to implement ([Barequet and Har-Peled 2001; Ericson 2004]).

Most of the time, heuristic approaches are used instead. The most popular ones are based either on principal component analysis [Jolliffe 2002] or on brute-force search. Note that given one axis $\mathbf{p}$ of an optimal OBB, the two remaining axes can easily be obtained by using the rotating calipers method to compute the minimal area rectangle enclosing the set of points projected on a plane orthogonal to $\mathbf{p}$. Several methods use this idea of finding the orientation of the OBB, aligned with a given direction $\mathbf{p}$, that has the minimal volume by solving the associated 2D problem.

In this section, these algorithms are presented, but a more detailed discussion can be found in [Ericson 2004]. We have implemented them all with Matlab® in a unified framework in order to compare these currently widespread methods for OBB fitting. Several studies comparing the performances of bounding box algorithms can also be found in [Dimitrov et al. 2009a; Lahanas et al. 2000].

2.1 O'Rourke's algorithm

In [O'Rourke 1985], Joseph O'Rourke presented an algorithm that can be used to compute an optimal OBB of a set of points in 3D. Although exact, this method has the main drawbacks of being both extremely complicated and very slow. It can be seen as a generalization of the rotating calipers for the 3D case. Indeed, it is a smart exhaustive search across all potential optimal orientations of the bounding box. Thus, this algorithm is based on a property that must be satisfied by the minimal OBB, and which is stated in the following theorem:

THEOREM 1 [O'ROURKE 1985]. *A box of minimal volume circumscribing a convex polyhedron must have at least two adjacent faces that contain edges of the polyhedron.*

These two faces are said to be *flush* with edges.

Based on this property, O'Rourke then devised an algorithm that examines every pair (e_1, e_2) of edges of $\mathcal{CH}(\mathcal{X})$. The idea is to perform a rotation of the OBB such that a face and an adjacent one are continuously flush with e_1 and e_2, respectively. Such a rotation is shown in Fig. 2. The volume of the OBB is a continuous but non-smooth function of the rotation matrix R. Indeed, the derivative is not continuous each time a third edge is flush with one face of the OBB. Between two such particular rotations, the volume is a rational function whose local minima can be obtained from the roots of a polynomial of degree 6. If one of these volumes or the volume with three flush edges is smaller than the current best volume found, the incumbent optimal solution is updated.

This algorithm runs in cubic time since the computational cost for each pair of edges is linear in N_V, as in the 2D rotating calipers

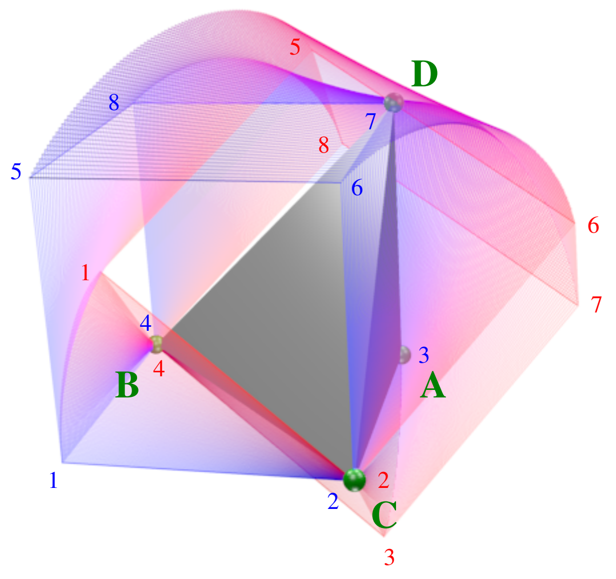
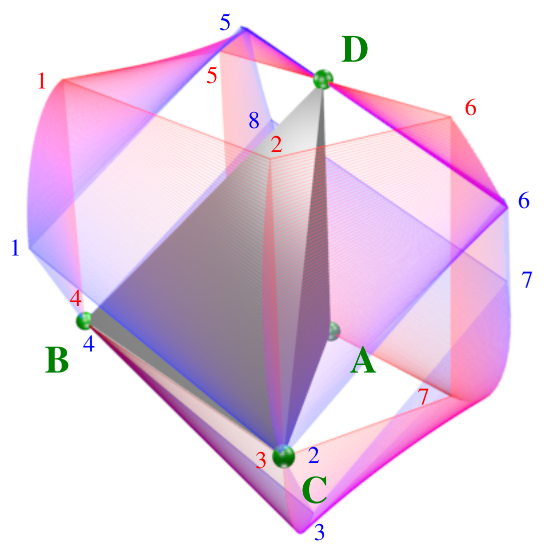


Fig. 2. Illustration of two successive steps of the rotation involved in O'Rourke's algorithm for the tetrahedron $ABCD = \{(0,0,0),(1,0,0),(0,1,0),(0,0,1)\}$. During the whole rotation, the adjacent faces 3784 and 1234 are flush with the edges AB and BC, respectively. Hence, both faces evolve in a plane rotating around their corresponding edge. In general, the rotation of the other faces are not around a fixed axis as adjacent faces of the OBB have to stay orthogonal while being flush with at least one vertex of the tetrahedron. The first OBB is the cube in blue of which the faces 1234, 3784 and 2376 are flush with the faces ABC, ADB and ADC of the tetrahedron, respectively. In fact, the latter is a corner of this cube; hence, all edges are flush with faces of this particular OBB. In the intermediate bounding box, which is in red (resp. blue) on the left (resp. right), the face 1256 is flush with the edge CD; since this particular OBB has three edges flush, it corresponds to a salient point of the volume function during the rotation. As far as the last bounding box, in red on the right, is concerned, the faces 3784 and 5678 are flush with the face ABC and the edge AD, respectively. Note that its vertex 8 is hidden behind the tetrahedron.



technique, and there are $\mathcal{O}(N_V^2)$ pairs of edges. We have implemented O'Rourke's algorithm in order to compute the optimal volume and thus verify the accuracy of the other methods.

2.2 PCA-based methods

A very popular class of heuristic methods is the one based on principal component analysis. The idea behind it is to first perform a PCA on $\mathcal{X}$, that is, computing the eigenvectors of its covariance matrix and choosing them as axes of the orthonormal frame $\mathbf{e}_\xi$. The first (resp. last) axis is the direction of largest (resp. smallest) variance. Either some or all axes resulting from the PCA can then be used, the choice leading to three different variants of the methods [Barequet et al. 1996].

2.2.1 *All-PCA.* This is the most basic method. It consists in directly using the three directions given by the PCA as the axes of the OBB. In practice, $\mathcal{X}$ is simply rotated to be placed in the PCA frame, and the AABB in this frame is obtained by computing the minimum and maximum for each component.

Though this method is particularly fast and easy to implement, it can be shown that the ratio between the volumes of the OBB obtained using PCA and the optimal volume is unbounded [Dimitrov et al. 2009b]. In fact, the PCA is very sensitive to the way points are distributed, and can fail to give good results even in simple cases.

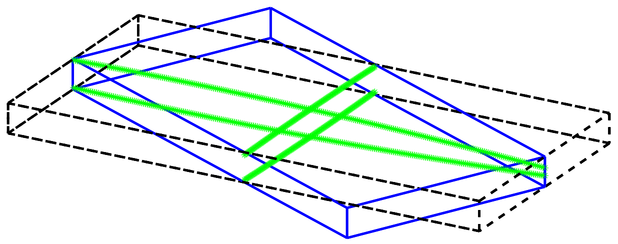


Fig. 3. Simple case showing the OBBs given by All-PCA or Max-PCA (in black), and Min-PCA or Continuous PCA (in blue).

For example, the PCA yields a very badly fitted bounding box if the points form two crosses roughly on two parallel rectangles, as shown in Fig. 3. This is why one of the two following variants are often used to improve the quality of the solution.

2.2.2 *Max-PCA and Min-PCA.* The idea here is to use only one of the three axes determined by the PCA. The orientation of the OBB, aligned with this axis, that has the minimal volume can then be obtained by solving the associated 2D problem obtained by projecting the points on the plane formed by the two unused axes. In the Max-PCA variant, the direction of projection is given by the first axis which is the eigenvector corresponding to the largest eigenvalue of the covariance matrix. On the other hand, one can choose to keep the last axis instead and thus obtain the Min-PCA variant.

These methods are a little slower, but are always more accurate than All-PCA and may avoid some of its pitfalls. For example, using Min-PCA (resp. Max-PCA) will give an almost optimal bounding box if $\mathcal{X}$ has a predominantly two-dimensional (resp. one-dimensional) shape. Fig. 3 illustrates the case of an object that has one dimension with a significantly much smaller range than the others. On sets of points with a real 3D extension, the Min-PCA variant tends to yield better results in practice.

2.2.3 *Continuous PCA.*

As regular PCA-based algorithms are too sensitive to the distribution of the points in the set, Stefan Gottschalk, Ming Lin and Dinesh Manocha have proposed the following two improvements in [Gottschalk et al. 1996]:

(1) Only the vertices of $\mathcal{CH}(\mathcal{X})$ determine the optimal bounding boxes. Hence, computing the PCA simply on $\mathcal{X}^C$ should provide better results.
(2) This reduced set of points can still give poor results; it typically happens when points are very concentrated in one region of the convex hull. A resampling of $\mathcal{CH}(\mathcal{X})$ should be performed in order to obtain a uniformly spread out set.

Those two improvements yield a slightly more complex method because, to be completely general, the convex hull needs to be resampled “infinitely densely”; this leads to a reformulation of the covariance matrix as a continuous form. However, it is shown in [Dimitrov et al. 2009b] that, for an octahedron, the volume of the bounding box given by this method is still four times bigger than the optimal volume.

2.3 Brute-force methods

Given the complexity of the problem, a common simple strategy is to examine a large number of possible orientations and choose the one yielding the best OBB. These methods are used quite regularly given the ease of implementation. Some of them are independent of the distribution of points in the geometry, unlike PCA-based heuristics.

There are two principal classes of heuristics. First, one can decide to sample the search space and try all these possible orientations. For example, a uniform distribution can be used to sample $SO(3,\mathbb{R})$, and then try all these rotation matrices. Another possibility is to consider a large set of points on the unit sphere, each point defining a direction $\mathbf{p}$. The orientation of the OBB, aligned with p, that has the minimal volume can then be obtained by solving the associated 2D problem. In either case, the best OBB obtained can be quite close to the real optimum provided that the number of considered orientations is sufficiently large. Obviously, the major drawback of this approach is the large computation time required to check all these orientations. More details can be found in [Ericson 2004].

A second approach is to select a large set of candidates based on some properties of the geometry. For every direction that connects any two points of the set, the associated 2D problem can be solved to obtain the orientations of the OBB. The one that yields the smallest volume is then selected. This heuristic is described in [Barequet and Har-Peled 2001] as the *all-pairs* method.

Many other strategies can be used. In [Korsawe 2008], three variants are described. The fastest one was already proposed in the discussion section of [O'Rourke 1985] and corresponds to a search among all bounding boxes of which one side coincides with one face of the convex hull. It is thus a generalization of the 2D naive $\mathcal{O}(N_V^2)$-time algorithm. This algorithm was based on the theorem which states that each optimal oriented bounding rectangle has one side coinciding with one edge of $\mathcal{CH}(\mathcal{X})$ [Freeman and Shapira 1975]. Both other strategies also take edges of the convex hull into account but are slower.

2.4 (1+ε)-approximation

In [Barequet and Har-Peled 2001], Gill Barequet and Sariel Har-Peled have proposed algorithms to compute, for any value of $\varepsilon \in]0,1]$, an approximating OBB whose volume is at most $(1+\varepsilon)$ times the optimal volume. For that purpose, a grid whose discretization step d is inversely proportional to ε is built. Its orientation is given by an OBB whose volume is a constant factor approximation of the optimal one. This OBB is chosen aligned with an approximation of the diameter of $\mathcal{X}$ that is computed using the AABB.

Based on this grid, two very different methods can be designed. On the one hand, $\mathcal{X}$ can be projected on the grid with a computational cost that is linear in N. O'Rourke's algorithm can then be performed on the projected set. In addition to requiring an implementation of O'Rourke's method, a main drawback of this method is that it can be very slow in practice if d is too large. This first method is called APPROXMINVOLBBX. On the other hand, each vector pointing from the center of the grid to one of its nodes can be taken as a direction $\mathbf{p}$. The orientation of the OBB aligned with $\mathbf{p}$, that has the minimal volume, can then be obtained by solving the associated 2D problem. For this variant, it is necessary to choose how many cells of the grid will be considered to build a candidate OBB. This is done by considering only the cells that have a Chebyshev (or infinity-norm) distance of d or less from the center of the grid. The way to compute the value of d to reach an accuracy of ε is exposed in [Barequet and Har-Peled 2001]. This second method is referred to as GRIDSEARCHMINVOLBBX in the article.

Note that the original technique described in [Barequet and Har-Peled 2001] can be improved. Indeed, it is possible to reduce the size of the projected set of points to a smaller set as described in [Agarwal et al. 2004; Chan 2006].

3. A NEW METHOD BASED ON OPTIMIZATION

The approach used in this article consists in formulating the search of the minimal volume OBB as an optimization problem defined on a manifold. To the best of our knowledge, this idea has only been used in [Lahanas et al. 2000]. In that article, the authors combine Powell's quadratic convergent method [Press et al. 1992] and a multi-scale grid search to minimize the volume of the OBB. A comparison of this scheme with our algorithm can be found in section 3.3. The method we propose is based on a formulation of the minimal OBB problem as an unconstrained optimization problem on $SO(3,\mathbb{R})$, which is presented below.

3.1 Formulation of the optimization problem

A first direct optimization formulation can be written as follows:

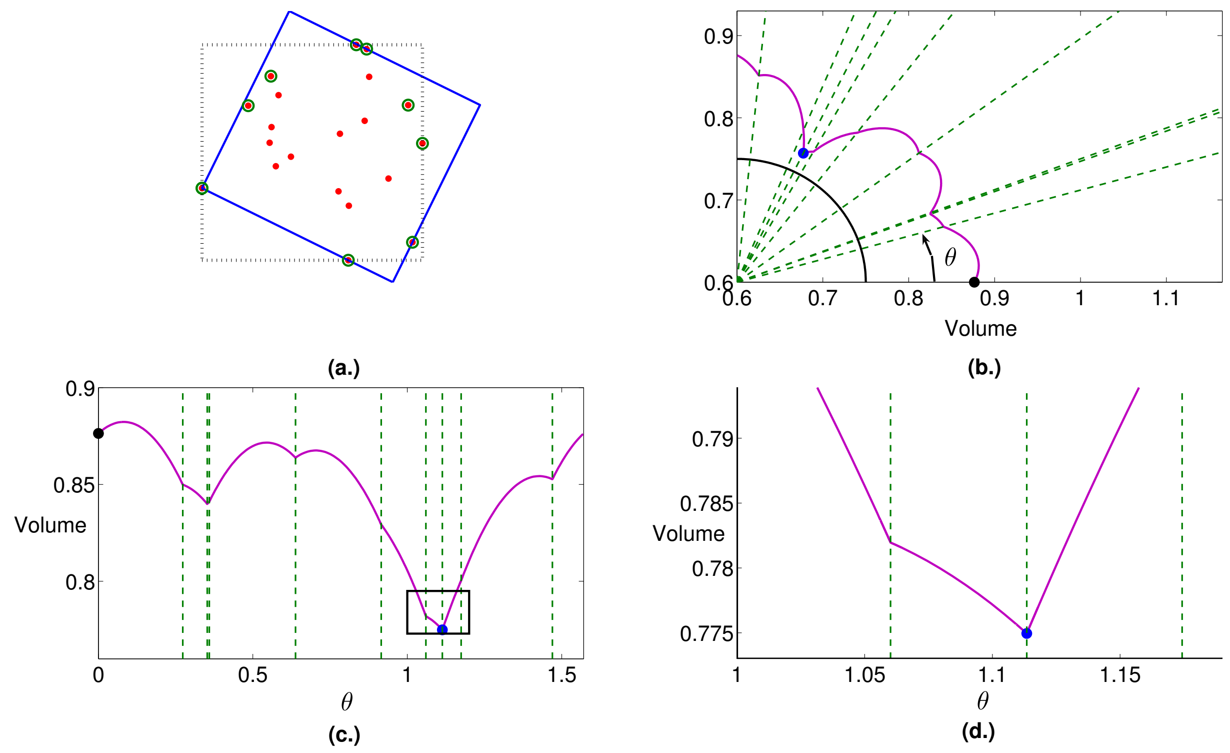


Fig. 4. **(a.)** A 2D example is shown with the same conventions as in Fig. 1. **(b.)** The volume of the minimal oriented bounding rectangle is shown for all directions around a quarter of the circle. This is sufficient given the symmetries of a rectangle. Salient points corresponding to angles aligning the bounding rectangle with a face of $\mathcal{CH}(\mathcal{X})$ are shown by dashed lines.**(c.)** The same curve is also drawn as a function of the angle θ ranging from 0 to $\frac{\pi}{2}$. It appears that only four of the salient points are local minima. **(d.)** A close-up view on the global minimum is shown.

$$\begin{aligned} \min_{\substack{\mathbf{\Delta},\mathbf{\Xi}\in\mathbb{R}^3\\ R\in SO(3,\mathbb{R})}} \quad & \Delta_\xi\Delta_\eta\Delta_\zeta \\ \text{s.t.} \quad & -\frac{\mathbf{\Delta}}{2} \le R\mathbf{X}_i - \mathbf{\Xi} \le \frac{\mathbf{\Delta}}{2} \qquad \forall i\in\{1,\dots,N\}, \end{aligned} \tag{1}$$

where $\mathbf{X}_i \in \mathcal{X}$, $\mathbf{\Delta} = (\Delta_\xi, \Delta_\eta, \Delta_\zeta)$ denotes the dimensions of the OBB and $\mathbf{\Xi}$ its center after rotation by R. Note that the $\le$ operator is applied componentwise.

The objective function is trilinear and the constraints are linear. One of the major difficulties in this problem is the feasible set of R. The minimization over those two vectors in $\mathbb{R}^3$ and this rotation matrix can be carried out as two successive minimizations. One can single out the minimization with respect to R, and the other "internal" one is simply given by the AABB after a rotation defined by a given fixed R of $\mathcal{X}$. Thus, the problem (1) can also be written in the following form:

$$\min_{R\in SO(3,\mathbb{R})} f(R), \tag{2}$$

where the objective function $f(R)$ is simply the volume of the AABB of $\mathcal{X}$ rotated by R:

$$f(R) = \left(\begin{array}{ll} \min\limits_{\mathbf{\Delta},\mathbf{\Xi}\in\mathbb{R}^3} & \Delta_\xi\Delta_\eta\Delta_\zeta \\ \text{s.t.} & -\frac{\mathbf{\Delta}}{2} \le R\mathbf{X}_i - \mathbf{\Xi} \le \frac{\mathbf{\Delta}}{2} \;\; \forall i\in\{1,\dots,N\} \end{array} \right).$$

Note that solving the problem (1) might prove to be more efficient since it is a relaxation of the problem $\min_{R\in SO(3,\mathbb{R})} f(R)$.

This function $f(R)$ is only $\mathcal{C}^0$ since it is not differentiable at every rotation matrix R that brings at least one face of the OBB to be flush with one edge of the convex hull. These particular rotations are the equivalent in $SO(3,\mathbb{R})$ of salient points and potentially yield local minima of this objective function. A 2D example illustrating the function $f(R)$ is presented in Fig. 4. It appears that this function is formed by the upper envelope of concave functions. Note that the problem (2) is also interesting in the 2D case, even though the rotating calipers technique already provides an efficient — and asymptotically optimal — linear algorithm.

Because of the non-differentiability of the function, especially at local minima, line search methods, such as Steepest Descent or Newton, can encounter convergence issues. Therefore, derivative-free methods have been preferred. Two main families of such algorithms can be distinguished.

The first one consists of the derivative-free optimization (DFO) methods that are guaranteed to converge to a local minimum. Three classes of DFOs can be distinguished. The Nelder-Mead simplex algorithm (see [Nelder and Mead 1965]) is one of the best known methods of optimization by direct search [Conn et al. 2009] and has led to many variants. A survey about direct search algorithms can be found in [Kolda et al. 2003]. The two other classes are derivative-free line-search methods (see [Kelley 1999]) and trust-region methods ([Conn et al. 2000]). A more detailed review on DFO methods can be found in [Conn et al. 2009].

The second family consists of the metaheuristics that try to explore the whole search space in order to find a global optimum. However, they often have a rather slow convergence and might even fail to converge. Some well-known examples are simulated annealing (see [Kirkpatrick et al. 1983]), tabu search (see [Glover 1989; 1990]), or genetic algorithm ([Goldberg 1989; Holland 1975]).

In order to find a global minimum of the volume function $f(R)$, a local search component to ensure the convergence to a minimum and a global search component to explore the search space can be combined. Several schemes are possible but in this article, we have chosen to focus on a hybrid method combining the Nelder-Mead algorithm and the genetic algorithm, as preliminary tests were showing very promising results. Another example of an applicable global-local scheme for this problem is the particle swarm optimization (see [Borckmans and Absil 2010]).

3.2 The Hybrid Bounding Box Rotation Identification algorithm

The first method used in HYBBRID is the Nelder-Mead algorithm. Its principle is simple: it starts with an initial simplex, i.e., a set of $d + 1$ affinely independent points, where d is the dimension of the search space. At each iteration, the worst point is replaced by its reflection through the centroid of the d remaining points, if this point is good enough. If this new point is even better than the best current point, a good search direction may have been found, so the algorithm tries to explore further in that direction. If the new point is not good enough, the optimal point may lie inside the simplex, so we shrink it. More details about these steps are shown in Fig. 5. This method is not guaranteed to converge to a stationary point (for example see [McKinnon 1999]), and largely depends on the initial simplex. Several runs with different initial simplices can be performed, or more elaborated variants can be used, in order to increase the probability of reaching a global minimum.

By combining Nelder-Mead simplex algorithm and genetic algorithm one could hope to obtain very good solutions in a short time. This idea has been studied by several authors: in [Chelouah and Siarry 2003] the authors first use a continuous genetic algorithm with elitist strategy to locate a "promising area" in the search space, where the Nelder-Mead algorithm is then used to try to find the best solution situated in that region. Another example is given in [Durand and Alliot 1999], where the authors apply genetic algorithm with p-tuples of points as population elements, with several Nelder-Mead iterations applied at each generation.

The combination used here is close to the variant of Durand and is detailed hereafter, step by step. As the dimension of the rotation group is $d = 3$, a simplex is a set of four rotation matrices $\mathcal{R} = \{R_1, R_2, R_3, R_4\} \subset SO(3, \mathbb{R})$. It forms a tetrahedron on this manifold with R_j at its vertices. An element A_k of the population $\mathcal{A}$ is a simplex $\mathcal{R}$ and its *fitness* is defined as $\min_{j\in\{1,\dots,4\}} f(R_j)$. The HYBBRID algorithm can be decomposed in the following steps:

(1) *Initialization.* Let M be the size of the total population. It is initialized with random simplices, i.e., the four vertices R_j of each simplex are obtained by QR factorization of random 3-by-3 matrices.

(2) *Selection.* The fitness of all the simplices is evaluated. The best $\frac{M}{2}$ simplices are selected, the others are discarded. From this reduced population, four groups $\mathcal{A}_1^I, \mathcal{A}_2^I, \mathcal{A}_1^{II}, \mathcal{A}_2^{II}$ are created at random using a uniform distribution. Each group has $\frac{M}{2}$ elements, and one population member can be in one group, several groups, or none, and can be selected any number of times in each group.

(3) *Crossover I.* A standard mixing crossover is applied between $\mathcal{A}_1^I$ and $\mathcal{A}_2^I$. A pair of parents is constituted by choosing the k^{th} element of both subpopulations: $A_1 \in \mathcal{A}_1^I$ and $A_2 \in \mathcal{A}_2^I$. They produce an offspring $A_{0,i}$. Each vertex of the simplex A_0 is either the corresponding vertex of A_1 or of A_2, the selection being random, but the parent with the best fitness having a higher probability of being chosen. This gives us $\frac{M}{2}$ new simplices.

(4) *Crossover II.* The other $\frac{M}{2}$ new simplices are given by an affine combination crossover between $\mathcal{A}_1^{II}$ and $\mathcal{A}_2^{II}$. Let $A_1 \in \mathcal{A}_1^{II}$, $A_2 \in \mathcal{A}_2^{II}$ be the k^{th} pair of parents as before. The four vertices $A_{0,j}$ of the corresponding offspring A_0 are defined by $A_{0,j} = \lambda A_{1,j} + (1 - \lambda)A_{2,j}$, where the value of λ depends on whether A_1 is better or worse than A_2. For example, λ can have the value 0.4 (resp. 0.6) if the fitness of A_1 is smaller than that of A_2.

(5) *Mutation.* K Nelder-Mead iterations are applied on all these M new simplices to obtain the new generation of the population.

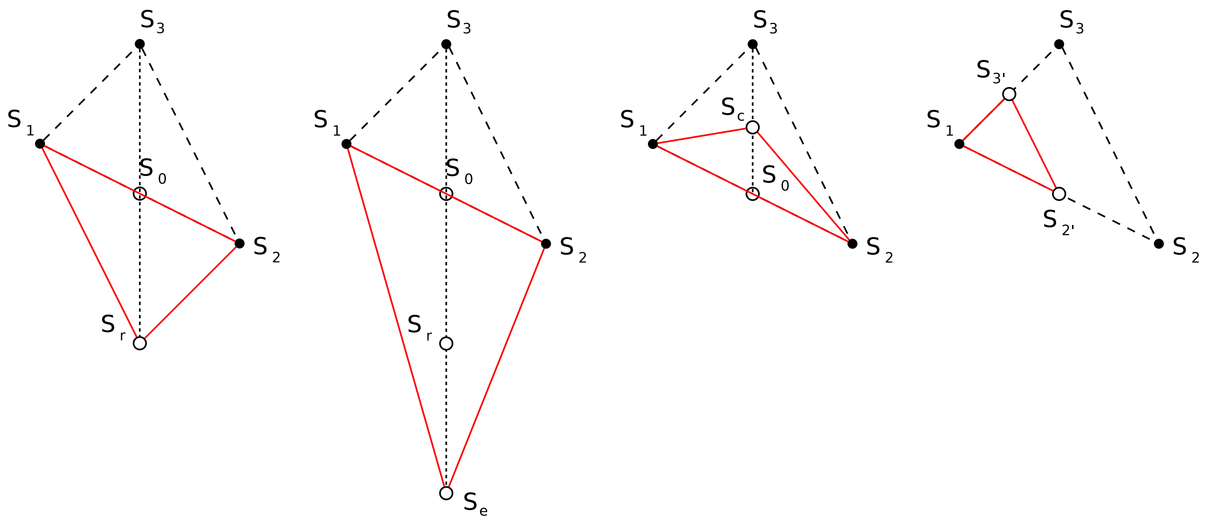
(6) *Stopping criterion.* This process (Selection – Crossover – Mutation) is repeated until a stopping criterion is met, usually if the fitness of the best simplex stalls for several iterations w.r.t. the desired tolerance, or if a maximal number of iterations is reached. In our case, the algorithm stops after k consecutive generations where the objective value does not improve by at least x % compared to the current best value, with $k = 5$ and $x = 1$ as default values for these parameters.

The goal of the genetic component of HYBBRID is to somehow compute the initial condition so that the Nelder-Mead algorithm converges to a global minimum. These steps, inspired by evolutionary biology, bring correlations between the initial conditions, which is better than random simplices.

Using Nelder-Mead simplices seems a better choice than directly considering rotation matrices with a mutation consisting in a line search method. Indeed, the use of Nelder-Mead simplices induces a layer of local cooperation between groups of candidate solutions. This meshes well with the "global" cooperation introduced by the genetic algorithm that randomly "resets" the initial conditions of the Nelder-Mead algorithm.

Experiments showed that using only one of the two crossover steps yields poorer performances. Combining both ensures that the simplices move enough to explore the search space sufficiently (crossover I) while still tending to gather around promising areas (crossover II). Of course, other crossovers could also be considered.

The crossover II and the update of the simplices in Nelder-Mead consist of affine combinations of rotation matrices. The geodesics can be used in order to take into account the geometry of $SO(3, \mathbb{R})$ in these computations. On this particular manifold, the exponential map and the log map [Krakowski et al. 2007] can be used. However, a high accuracy is not required in our Nelder-Mead method as only a few iterations are applied to let the population get closer to the minima at each generation. In this case, the method can take



(a.) Reflection **(b.)** Expansion **(c.)** Contraction **(d.)** Reduction

Fig. 5. Illustration of the Nelder-Mead algorithm in $\mathbb{R}^2$ to minimize a function $F(x,y)$. Let $S_1S_2S_3$ form a simplex $\mathcal{S} \subset \mathbb{R}^2$ such that $F(S_1) \leq F(S_2) \leq F(S_3)$. At each iteration, the worst point, S_3 here, is always removed. Let S_0 be the centroid of all remaining points, and S_r be the reflection of S_3 through this centroid. **(a.)** If S_r is better than the current worst remaining point, S_2 here, it is chosen to define the new simplex $S_1S_rS_2$. **(b.)** If S_r is even better than the current best point, namely S_1, the simplex is expanded in the direction S_0S_r, giving $S_eS_1S_2$ as the new simplex. The expansion is defined such that $S_e = S_r + (S_r - S_0)$. If the reflected point is worse than all of the current points, the algorithm tries to contract the simplex. **(c.)** If the contracted point, $S_c = \frac{1}{2}(S_0 + S_3)$, is better than S_3, then the new simplex is given by $S_cS_1S_2$. **(d.)** Otherwise, the whole simplex is reduced around the current best point, S_1. In this case, the new simplex is $S_1S_{2'}S_{3'}$, with $S_{i'} = \frac{1}{2}(S_1 + S_i)$ for all i.

advantage of the fact that $SO(3,\mathbb{R})$ is embedded in $\mathbb{R}^{3\times 3}$. After computing the affine combination in $\mathbb{R}^{3\times 3}$, the obtained matrix can be projected on $SO(3,\mathbb{R})$ with a QR factorization to obtain an approximation of the affine combination on this manifold [Sarlette and Sepulchre 2009]. The proper projection on $SO(3,\mathbb{R})$ is actually the polar factorization which is orthogonal for the inner product defining the Frobenius norm [Golub and van Loan 1996]. However, it requires to perform a singular value decomposition which is slightly more expensive. In practice, it is observed that using QR factorization is faster than polar decomposition without losing significant accuracy.

The reflection and expansion steps on the simplex $\mathcal{R}$ are combinations that are not convex. Nevertheless, such extrapolations can be avoided as in these particular cases, the geodesic on $SO(3,\mathbb{R})$ can be followed exactly. Indeed, on this manifold, the mathematical operation that brings R_3 on R_0 is the left multiplication by the rotation matrix $R_0R_3^T$ since $R_0 = (R_0R_3^T)R_3$. Let us consider the reflection step for instance. By definition, R_r is obtained by performing the same displacement on the manifold but starting from R_0 instead of R_3. Hence, the reflection and expansion points can simply be computed as $R_r = R_0R_3^TR_0$ and $R_e = (R_0R_3^T)^2R_0$, respectively. Similarly, the new vertices in the contraction and reduction step can be written in a closed form, e.g. $R_c = (R_0R_3^T)^{1/2}R_3$. However, the computation of the square root of a matrix is more expensive than the QR factorization. Hence, following the geodesics in the computation of R_r and R_e is interesting, which is not the case for the contraction and reduction of the simplex $\mathcal{R}$.

Finally, a post-processing step can be applied to the OBB obtained from any algorithm: as the 2D problem is easy to solve with the rotating calipers technique, the set $\mathcal{X}$ can be projected along one of the axes of the candidate OBB and solve the associated 2D problem. This amounts to a rotation of the box around the normal of one of the faces and ensures *local* optimality in that direction.

3.3 Comparison with the algorithm of Lahanas et al.

Unfortunately, it is difficult to empirically compare the efficiency of HYBBRID with the algorithm of [Lahanas et al. 2000] because the way Powell's method is applied and the choice of the parameters of the multi-scale grid search method are not detailed in the article. Nevertheless, let us emphasize the main differences from a theoretical point of view between these two hybrid approaches.

First of all, the formulation of the problem is not defined on the same search space. On the one hand, the principle of HYBBRID is to minimize an objective function on the rotation group $SO(3,\mathbb{R})$. Each evaluation of this cost function requires an AABB computation which is a trivial optimization subproblem. One important property of this method is that the search space is viewed as a manifold without a global parameterization. On the other hand, the search space on which the optimization problem is formulated in [Lahanas et al. 2000] is parameterized by the triplet $(\phi, \cos\theta, \alpha)$, i.e., the azimuth angle, the cosine of the zenith angle and the angle of rotation around the axis defined by ϕ and θ, respectively. A drawback of this choice is the singularity induced at the poles. One main interest of formulating the optimization problem on a manifold is to avoid such issues induced by the parameterization.

Both algorithms consist in a hybridization using an exploration and an exploitation component. The latter is the Nelder-Mead (resp. Powell) method for HYBBRID (resp. the algorithm of Lahanas et al.). On the one hand, the Nelder-Mead simplex search is an intuitive heuristic that is very popular due to its simplicity and empirical efficiency, at least for problems of dimension less than 5. This is the case here as the dimension of the $SO(3,\mathbb{R})$ rotation group is 3. At each iteration of this method, the vertices of the simplex induce an implicit model that is used to determine the next simplex with just a few evaluations of the cost function. On the other hand, Powell's method is a line-search method that requires solving a succession of one-dimensional minimization subproblems. As far as the exploration component is concerned, HYBBRID is based on the genetic algorithm which is stochastic while the multi-scale grid search method is a priori deterministic.

4. EXPERIMENTAL ANALYSIS

All the methods presented in the previous sections (except the method of Lahanas et al.) have been implemented using Matlab® and tested on about 300 sets of points from [GAMMA Group 2008]. These examples include a wide selection of different geometries, ranging from simple shapes to anatomical objects defined by millions of points. As a bounding box only depends on the convex hull of the object, computing it as a preprocessing step is a good way to reduce the number of points in the subsequent computations. The distributions of the number of points of the objects and of vertices on their convex hull, shown in Fig. 6, highlight the interest of such preprocessing. The characteristics of four of those examples are given in Table I, while a graphical representation is shown in Fig. 7. This figure is rendered in GMSH ([Geuzaine and Remacle 2009]), and the green meshes represent the convex hull of the objects.

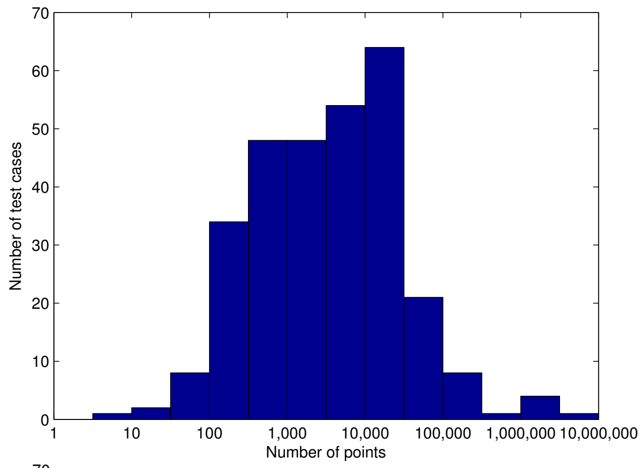
Note that for about 15 % of these objects, the AABB coincides with an optimal OBB. Furthermore, for about 40 % of the test cases, the AABB has at least one face parallel to a face of an optimal OBB.

In the remaining of this section, a study of the properties and behavior of HYBBRID is first presented in 4.1. This method is then compared to the different techniques introduced in section 2.

4.1 Performance of the Hybrid Bounding Box Rotation Identification method

The HYBBRID method was tested 200 times on each object of the test set presented in the previous section. HYBBRID was able to find an optimal OBB for each dataset. Nevertheless, this solution is not reached at each run because of the random component of the genetic algorithm. The actual success rate depends on parameters such as M and K. With carefully chosen values this success rate may be brought close to 100 %. However, changing the values of the parameters also influences the computation time required by the algorithm. Hence, we will first study the asymptotic time complexity of the algorithm in the following subsection. Then, we will analyze the effect of the two parameters M and K in terms of performance and computation time. Finally, the reasons why a suboptimal solution may be returned are explained as well as how it is possible to modify the algorithm in order to take these facts into account.

4.1.1 *Does it have a linear complexity?* One major drawback of O'Rourke's algorithm is its cubic time complexity, whereas the 2D problem can be solved with a linear complexity assuming the convex hull is known. An interesting point to investigate is thus



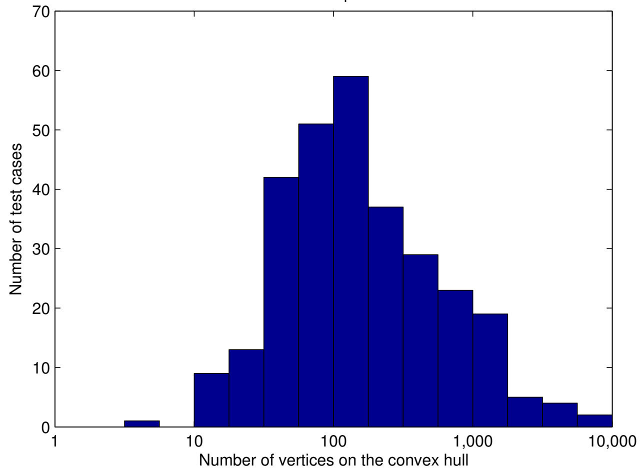
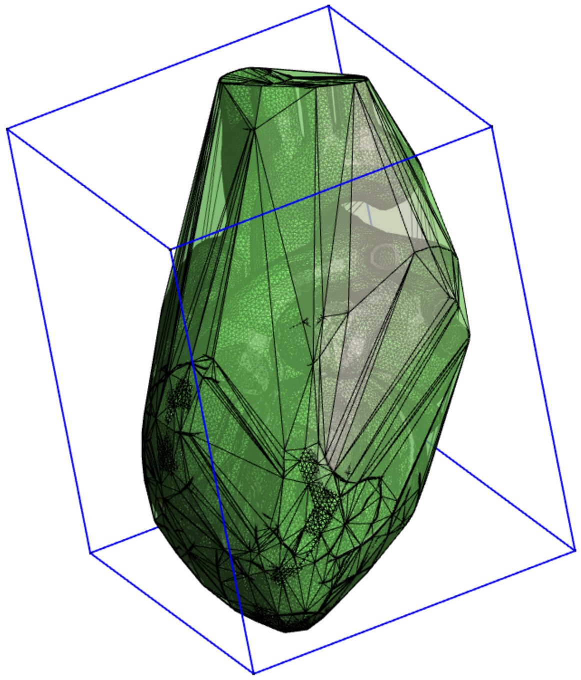


Fig. 6. Distribution of the number of points of the different objects and of vertices on their convex hull.

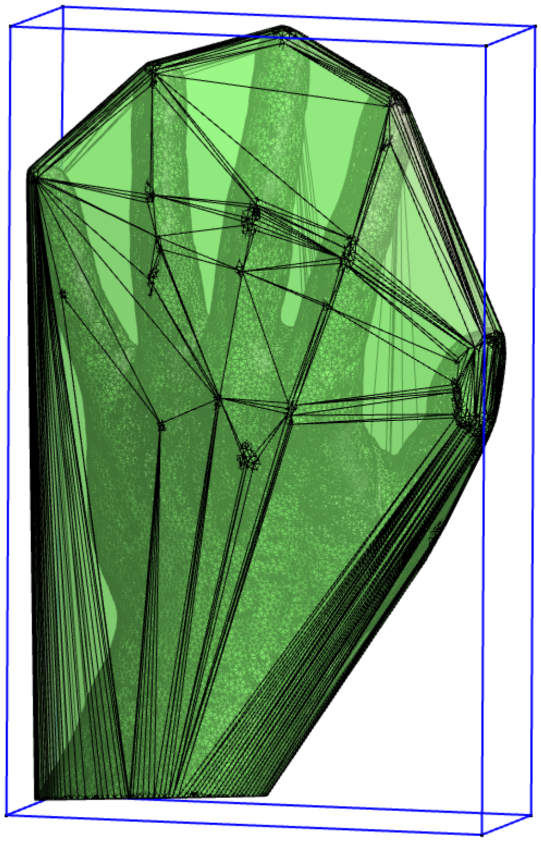
the time complexity of HYBBRID. In Fig. 8, the computation time needed by HYBBRID when run on each test case with $M = 30$ and $K = 10$ is represented. These experimental results tend to show that the asymptotic time complexity of this method would be linear with respect to the number of vertices on the convex hull N_V. Of course, this linear complexity does not include the convex hull computation done as a preprocessing step. The asymptotic linear complexity is illustrated by the red dotted line that is obtained using a weighted linear regression; indeed, data corresponding to test cases of large size have been given more weight, as we are looking for an asymptotic behavior.

As each iteration of HYBBRID takes $\mathcal{O}(N_V)$ time, which is the complexity of evaluating the volume of an AABB, the observation that the asymptotic time complexity of this optimization method seems linear would imply that the number of generations required to produce a solution is independent of the set of points. Note also that taking another pair of parameters would change the computation times, but the asymptotic complexity is expected to be the same.

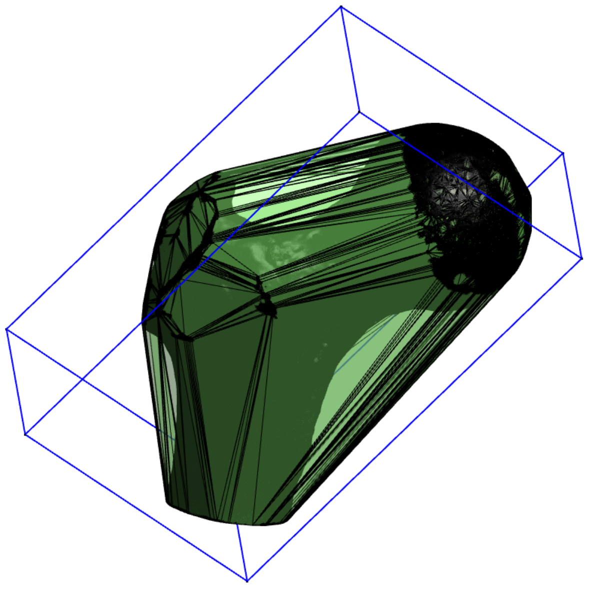
Theoretically, the total asymptotic time growing rate could be further lowered to $\mathcal{O}(\log N_V)$ by using the method described in [Edelsbrunner and Maurer 1985]. Unfortunately, the data structure described therein is difficult to implement in Matlab® due to limi-



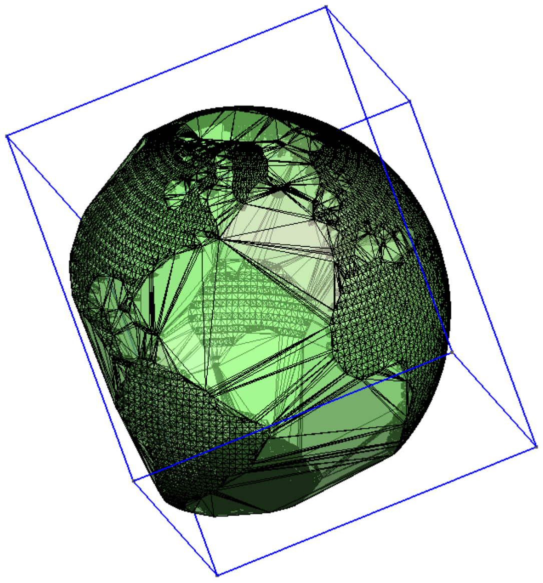
(a.) `heart482`



(b.) `hand770`



(c.) `balljoint4074`



(d.) `globe9306`

Fig. 7. Graphical representation of four examples of tested sets of points, along with their convex hull and the optimal OBB.

Table I. Characteristics of four examples of tested sets of points. The number in the name corresponds to N_V, the number of vertices of $\mathcal{CH}(\mathcal{X})$, and the datasets are ordered in increasing values of N_V. The second and third columns give the size of the original set of points and the time required to compute the convex hull, respectively. For the convex hull computation, the algorithm used is Qhull, which is written in C. The fourth column corresponds to the volume ratio for the minimal OBB. The fifth and sixth columns show the median and the maximal volume ratio respectively, obtained by considering 10000 randomly oriented bounding boxes. All computations have been carried out using Matlab® 7.6.0 (R2008a) on an Intel® Core™ 2 Duo 2.80 GHz with 3 GiB RAM, running Ubuntu Linux 10.04.

Name	N	Computation time of $\mathcal{CH}(\mathcal{X})$	$Volume(OBB)/Volume(\mathcal{CH}(\mathcal{X}))$		
			Optimal OBB	Random OBBs (median)	Random OBBs (max)
`heart482`	88608	0.1227 s	2.0070	2.5853	3.0226
`hand770`	47590	0.0630 s	2.1323	4.9615	6.6021
`balljoint4074`	137062	0.2337 s	1.9387	2.9988	3.6935
`globe9306`	19568	0.1108 s	1.8057	2.1185	2.3138

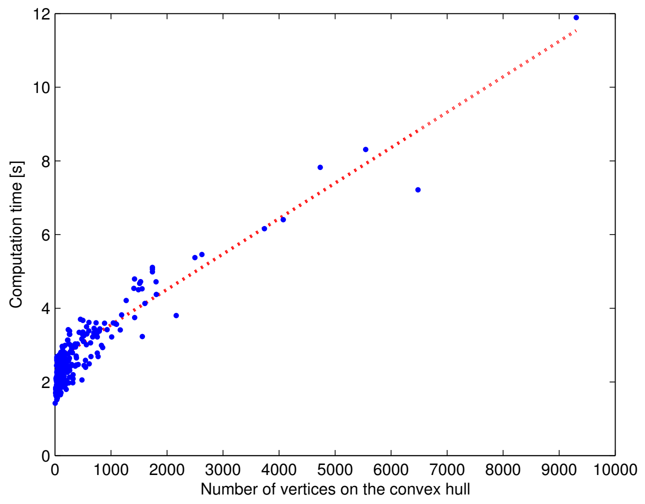


Fig. 8. Evolution of the computation time required by HYBBRID with data sets of increasing sizes, excluding the time required to compute the convex hull. Computer specifications are identical to what is described for Table I. The results were obtained by running the method 200 times on each test case and averaging the computation times.

tations in the usage of pointers in the language. Moreover, given the size of the test cases used, it seems that using this extreme points locating technique would not improve the total computation time in practice, as the overhead introduced by the construction of the structure would be too high.

4.1.2 *How do the parameters influence the optimality?* In Fig. 9, the reliability of our algorithm is studied through the performance and the computation time of HYBBRID for different sets of parameters. The performance is measured by the proportion of runs where the optimal volume has been obtained up to an accuracy of 10^{-12}. It is possible to check this as the exact optimal volume is known, thanks to O'Rourke's algorithm. Each example has been tested 200 times in order to take into account the variability due to the randomness inherent to HYBBRID. The stopping criterion used in these experiments was the following: "If there is no improvement of at least 1 % during at least 5 iterations, then the search is stopped".

It appears that increasing the number of Nelder-Mead iterations K yields the same general trend as increasing the population size M: this increases both the reliability and the computation time. Because of this natural trade-off, the optimal choice of parameters depends on the requirements for each particular application. For a given population size, the performance increases significantly with the number of Nelder-Mead iterations until about $K = 20$. As expected, for very small values of K such as 0 or 1, the performance is very weak. Indeed, this corresponds to a nearly pure genetic algorithm, whereas the main idea of HYBBRID is to keep locally improving the candidates found by exploring the search space, and repeatedly combine these improved solutions. After this threshold value of 20, the performance gain seems much less interesting, especially for large population sizes. Hence, one interpretation of this figure is that using 20 Nelder-Mead iterations is somehow optimal; the population size can then be chosen depending on the needs. As expected, the performance gain obtained by increasing the population size is more significant for small values of M: increasing $M = 10$ by 10 units is equivalent to an increase of 100 %, whereas going from $M = 40$ to 50 is only an increase of 25 %.

To summarize, based on these experimental results, we suggest taking $K = 20$ and then choosing M depending on the available time and the desired reliability. Note that if parallelization is considered for HYBBRID, one should take into account the fact that the genetic component is easily parallelizable whereas at each generation and for each population member, the Nelder-Mead iterations have to be applied sequentially. Hence in this case, it may be less expensive to increase the size M of the population than the number K of Nelder-Mead iterations and another value of K may be more appropriate.

4.1.3 *Why is it not always optimal?* Two main reasons may account for the fact that HYBBRID may sometimes miss the optimal volume (for example with our set of test cases, in about 4 % of the runs on average with the pair of parameters $(M, K) = (50, 30)$).

On the one hand, the algorithm consists in exploring the search space thanks to the evolution of a population of simplices. If the search is interrupted too early, a good exploration cannot be ensured; hence, a suboptimal solution may be returned. As there is no simple way to verify whether a given candidate bounding box is optimal, a time-accuracy trade-off has to be made with the stopping criterion. We have chosen a stopping criterion of the form: "If there is no improvement of at least x % during at least k iterations, then the search is stopped". Typical values used in the experiments are $x = 1$ and $k = 5$. Increasing k for example would increase

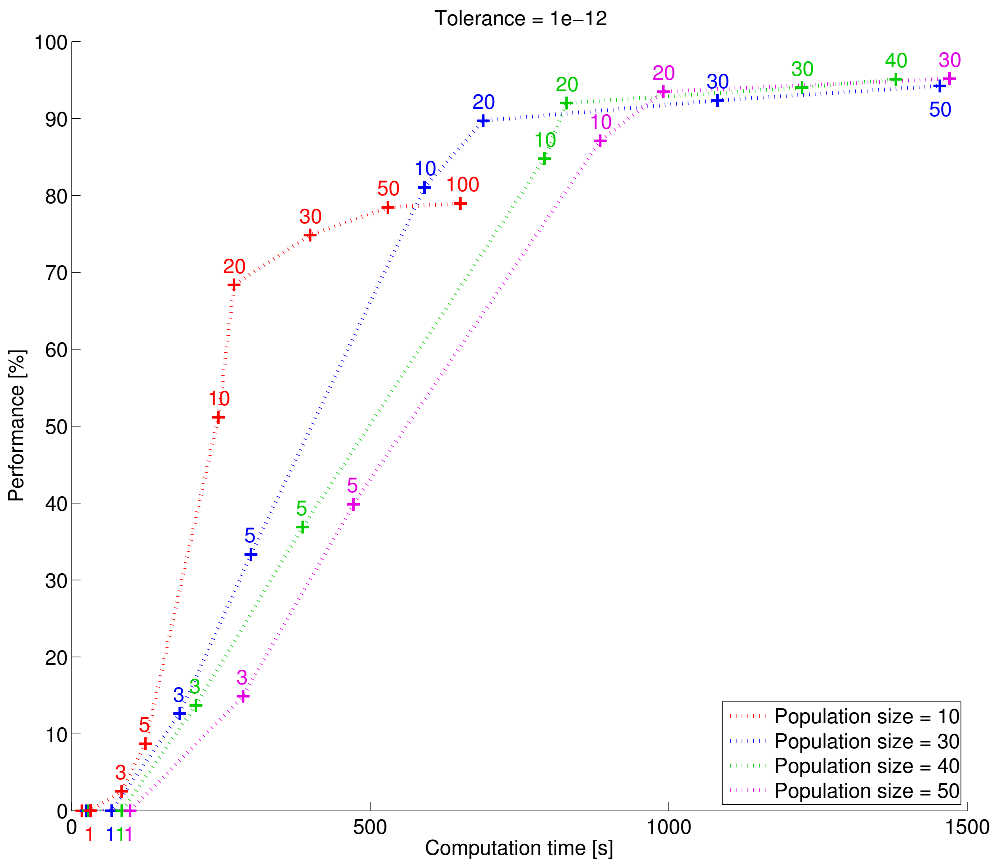


Fig. 9. Computation time and performance of the HYBBRID algorithm for different sets of parameters (M, K). The color of the points corresponds to the population size, whereas the numerical labels correspond to the value of Nelder-Mead iterations at each generation. The computation time corresponds to the total time required to approximate the OBBs of all 300 objects, including the computation of the convex hulls. For each test case, the results were obtained by running the method 200 times and averaging the results.

the expected number of iterations and thus the exploration of the search space. Many different schemes are possible, and the choice between a faster algorithm and more exploration should be made depending on the application.

On the other hand, it is possible that all simplices get stuck in a local minimum after some iterations. This behavior is desired if the minimum is global, but this may sometimes happen with sub-optimal solutions. One way to avoid such a situation could be to introduce random mutations, e.g., at each iteration, one population member is replaced by a random simplex. Another possibility is to apply random perturbations on some or all simplices with a given small probability. The choice of such a random mutation strategy is again a trade-off problem as using too many random mutations may reduce the effect of the improvements given by the Nelder-Mead algorithm. Conversely, with a small random mutation factor, the simplices may remain at a suboptimal solution for too long, possibly activating the stopping criterion. In this work, we have chosen not to use such random mutations for the sake of simplicity, but this strategy can be easily included in the algorithm. Moreover, the results show that the correct bounding box is found in nearly all cases, and at least a good solution is found in all cases, as it can be observed in the figures in the previous subsection.

4.2 Comparison of HYBBRID to the state of the art

In this section, the proposed algorithm is compared to the other methods described in section 2. First, observations are made on a few very simple examples to highlight some of the strengths and shortcomings of the different techniques. Then, all the methods are compared on the whole set of test cases to extract more information about how they compare in terms of computation time and reliability.

4.2.1 *What are the methods' basic properties?* The different methods with some of their general characteristics are presented in Table II. It appears that HYBBRID is the only iterative method in the table. This is an advantage as iterative methods tend to be more robust than direct ones. Note that only all-pairs, Korsawe's and O'Rourke's methods are not linear with respect to N_V; the first two being much easier to implement than the third.

The computation of the convex hull is optional for HYBBRID. Indeed, keeping only the points on the convex hull does not change the result returned by HYBBRID but it may reduce the total computation time depending on the values of N and N_V. In the following results, the computation of $\mathcal{CH}(\mathcal{X})$ will be done as this gives a faster algorithm in nearly all test cases. The situation is differ-

Table II. General characteristics of the methods. Columns 2 and 3 show the type and the accuracy of the methods. The fourth column gives the worst-case asymptotic time complexity of the algorithms, in terms of the number of points N in the set, and the number N_V of vertices of the convex hull. The computation of $\mathcal{CH}(\mathcal{X})$ (with complexity $\mathcal{O}(N \log N)$, not included in the term in the fourth column) is needed for methods labeled with $\star$. It is not necessary but generally recommended for HYBBRID, which is indicated by the label $\star\star$. The last column indicates the approximate number of lines of code of the implementation.

Method	Category	Accuracy	Complexity	Implementation
O'Rourke	Direct, enumeration	Guaranteed exact	$\mathcal{O}(N_V^3)^\star$	$\sim$ 500 lines
HYBBRID	Iterative, optimization	Often exact in practice, parametric	$\mathcal{O}(N_V)^{\star\star}$ (experimental)	$\sim$ 400 lines
GRIDSEARCHMINVOLBBX	Direct, enumeration	$(1+\varepsilon)$-approximation	$\mathcal{O}(\frac{N_V}{\varepsilon^3})^\star$	$\sim$ 100 lines
PCA	Direct	Suboptimal	$\mathcal{O}(N)$	$\sim$ 100 lines
Gottschalk et al.	Direct	Suboptimal	$\mathcal{O}(N_V)^\star$	$\sim$ 20 lines
All-pairs	Direct, enumeration	Suboptimal	$\mathcal{O}(N_V^3)^\star$	$\sim$ 100 lines
Korsawe	Direct, enumeration	Suboptimal	$\mathcal{O}(N_V^2)^\star$ or $\mathcal{O}(N_V^3)^\star$	$\sim$ 200 lines

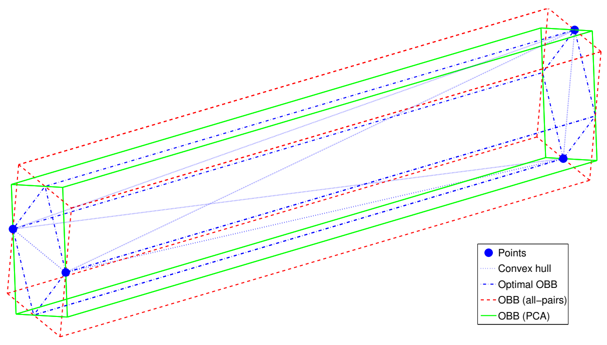


Fig. 10. Bounding boxes obtained by several methods, for the set S after rotation. O'Rourke's algorithm, HYBBRID, PCA/Gottschalk et al.'s method with post-processing and a variant of Korsawe's one are optimal (blue box). The all-pairs method gives the red suboptimal box that is two times larger, as does another variant of Korsawe's algorithm. The green box is obtained using PCA, but without post-processing.

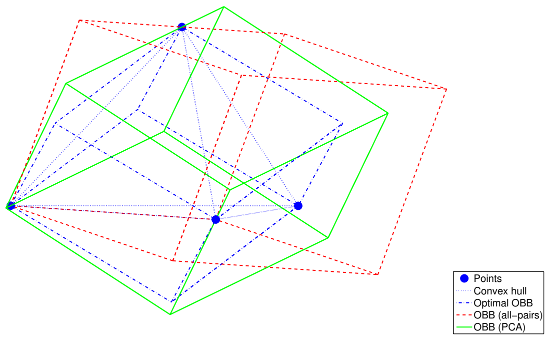


Fig. 11. Bounding boxes obtained by several methods, for the tetrahedron T after rotation. The optimal bounding box, in blue, is only obtained by O'Rourke's algorithm, HYBBRID and one variant of Korsawe's method. The red box can be obtained by using all-pairs method, whereas Min-PCA gives the green one.

ent with All-PCA, Max-PCA and Min-PCA as taking the convex hull of the set of points may result in a different OBB which is not always smaller.

The number of lines specified in the table is roughly the number of Matlab® code lines that were specifically written to implement the method. That is why some methods have a particularly low number of lines, for example Gottschalk et al.'s one. Using another language, where more low-level numerical operations would have to be either implemented from scratch or imported from a library would require more code. For other methods, namely PCA and Korsawe's, the number shown is for the whole set of variants. For example, writing a dedicated Matlab® script for All-PCA would not require more than ten lines of code.

Note that both HYBBRID and Barequet & Har-Peled's algorithm are methods with parameters (M, K for HYBBRID and ε (or equivalently d) for Barequet & Har-Peled). It is thus possible to obtain several compromises between accuracy and computation time, depending on the application.

It is also interesting to investigate which methods are exact on very simple examples. A first example is given by an arbitrary rotation of the following set of points:

$$S = \{(-1,-0.1,0),(-1,0.1,0),(1,0,-0.1),(1,0,0.1)\},$$

which is taken from [Barequet and Har-Peled 2001]. The bounding boxes obtained by several methods are shown in Fig. 10. Several observations can be made with regard to this example. First of all, O'Rourke's algorithm and HYBBRID are able to compute the optimal OBB. Conversely, most PCA-based methods are suboptimal without the post-processing, but all are optimal with it. In fact, the post-processing step rotates the bounding box around its three axes to try to find a better OBB. As at least one of the selected axes is correct in each case (the one corresponding to the main dimension of the dataset), a rotation around this axis is sufficient to obtain the optimal result. Note that Max-PCA is already optimal without this post-processing step since the main dimension of the dataset actually corresponds to the principal axis of the dataset in this case. A brute-force method like all-pairs, or naive variants of Korsawe's, are unable to find the correct solution. Indeed, the optimal OBB has no face orthogonal to an edge of $\mathcal{CH}(\mathcal{X})$, nor parallel to a face of the convex hull. One would need a more elaborate brute-force method to reach the optimum.

Concerning Barequet & Har-Peled's GRIDSEARCHMINVOLBBX method, the choice of the unit cell of the grid used in the algorithm influences the value of the parameter d that is required for a given accuracy. This unit cell is determined by an approximation of the diameter of $\mathcal{X}$, which is computed using the AABB. Hence, the grid and the performance of the algorithm depend on the initial orientation of the set of points. Even for this simple example rotated arbitrarily, GRIDSEARCHMINVOLBBX is

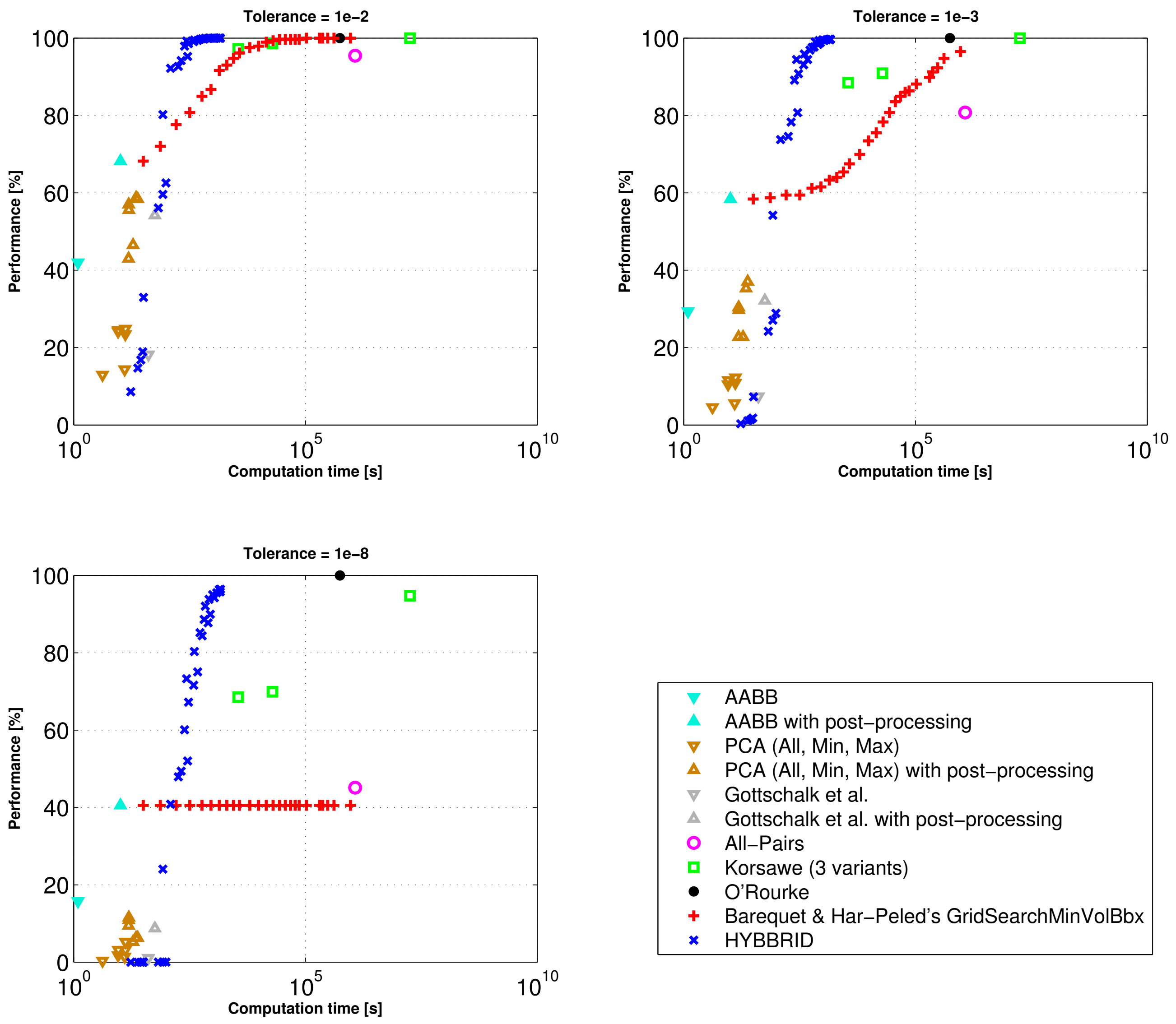


Fig. 12. Performance and computation time of all the algorithms. Computer specifications are identical to what is described for Table I. The computation time corresponds to the total time required to approximate an optimal OBB for each object among the 300 test cases, including the computation of the convex hulls if it is done by the algorithm. For HYBBRID, the volume of the OBB and the computation time of each test case have been obtained by running the algorithm 200 times and averaging the results.

only guaranteed to yield the optimal volume to any accuracy for sufficiently large values of d. For example, with the unrotated set S, GRIDSEARCHMINVOLBBX finds the optimal solution with a grid of size $d = 1$. However, with the rotated set $\widetilde{S}$ defined by:

$$\widetilde{S} = RS, \qquad R = \frac{1}{4}\begin{pmatrix} 0 & 2 & 2\sqrt{3} \\ 2\sqrt{3} & -\sqrt{3} & 1 \\ 2 & 3 & -\sqrt{3} \end{pmatrix},$$

GRIDSEARCHMINVOLBBX returns a solution whose volume has a relative error of 14 % with $d = 20$. Hence, although the algorithm will converge to the optimal value for $d \to \infty$, the orientation of $\mathcal{X}$ can significantly influence the quality of the results returned by this algorithm for finite values of d.

Another simple example is the tetrahedron obtained by a random rotation of the set:

$$T = \{(0,0,0), (0,1,1), (1,0,1), (1,1,0)\}.$$

Some results for this test case are shown in Fig. 11. Again, even if the geometry is also really simple, PCA methods fail to find the optimal bounding box, as do the all-pairs heuristic. Variants of Korsawe's method trying to build a bounding box aligned with faces of the convex hull are also bound to fail: as for the previous case, the tetrahedron is a geometry whose optimal bounding box is only flush with edges of said convex hull. The only methods that manage to find the optimal bounding box are O'Rourke's, HYBBRID and one of the variants of Korsawe's method. Note that this example illustrates the singularity of the PCA methods, which arises when the multiplicity of one of the eigenvalues of the covariance matrix is greater than 1. Then, the corresponding eigenvectors are not well-defined. In this case, this is due to the symmetry of the dataset.

4.2.2 *Where are they in the performance-computation time diagram?* All the methods have been tested on the whole set of test cases; the obtained results are shown in Fig. 12 for three accuracy values. This figure shows the computation time and the proportion of runs where the optimal volume has been obtained up to the given accuracy, as in Fig. 9. Each method or variant is represented by a particular point in this time-performance diagram. In fact, Fig. 9 corresponds to a close view of Fig. 12 without the logarithmic scale and displaying only the points corresponding to HYBBRID.

Barequet & Har-Peled's GridSearchMinVolBbx method has been tested with the following values for the grid size parameter: $d = 1, 2, \ldots, 9, 10, 12, 14, \ldots, 28, 30, 35, 40, 45, 50, 60$. If an accurate result is required, then the success rate (as defined in section 4.1.2) of the method stalls at about 40 % of the test cases for these values of d. Much higher values of d are thus needed in order to obtain an accurate result for most examples. Indeed, as $d \in \Theta(\frac{1}{\varepsilon})$, in order to improve the guaranteed accuracy by a factor k, one needs to increase the grid size d by a factor k. For instance, in order to reach a guaranteed accuracy of 10^{-8} instead of 10^{-3}, the value of d needs to be increased by a factor 10^5. However, if only a rough approximation such as 10^{-3} or less is required, this range of values of d may provide satisfactory results, as shown in Fig. 12. Note that to the best of our knowledge, Barequet & Har-Peled's algorithm was the only method providing such a trade-off between accuracy and computation time. Moreover, it appears that the values $d = 1$ and 2 yield smaller computation times than all other methods for their range of performance.

Another interesting observation is that the threshold value of 40 % corresponds to the performance obtained by using an AABB on which the post-processing step is applied. This can be explained by the symmetry and the natural definition of some objects in the test set, which are usually oriented in a way that is usual for human beings, and thus aligned with the principal axes. As such, for these objects, the AABB has a common axis with an optimal OBB. As GridSearchMinVolBbx is based on the AABB for the diameter approximation, the symmetry and the definition of the set of points (more precisely their ordering) also imply that the orientation of the grid has an axis aligned with one axis of an optimal OBB. Hence, solving the associated 2D problem will return an optimal solution.

As far as Korsawe's method is concerned, the two faster variants are either faster or more reliable than the other methods, except when compared to HYBBRID algorithm. The slower variant demonstrates very good performances, but unfortunately its computation time is much too large for practical purposes. The method is in fact even slower than O'Rourke's algorithm even if the latter has a guaranteed performance of 100 %. Another method enumerating a large set of directions is the all-pairs method. Unfortunately, its performance is dominated by Korsawe's, i.e., the latter appears to be both faster *and* more reliable than all-pairs.

As expected, PCA-based methods are very fast but provide solutions with a very limited accuracy. The quality may even be worse

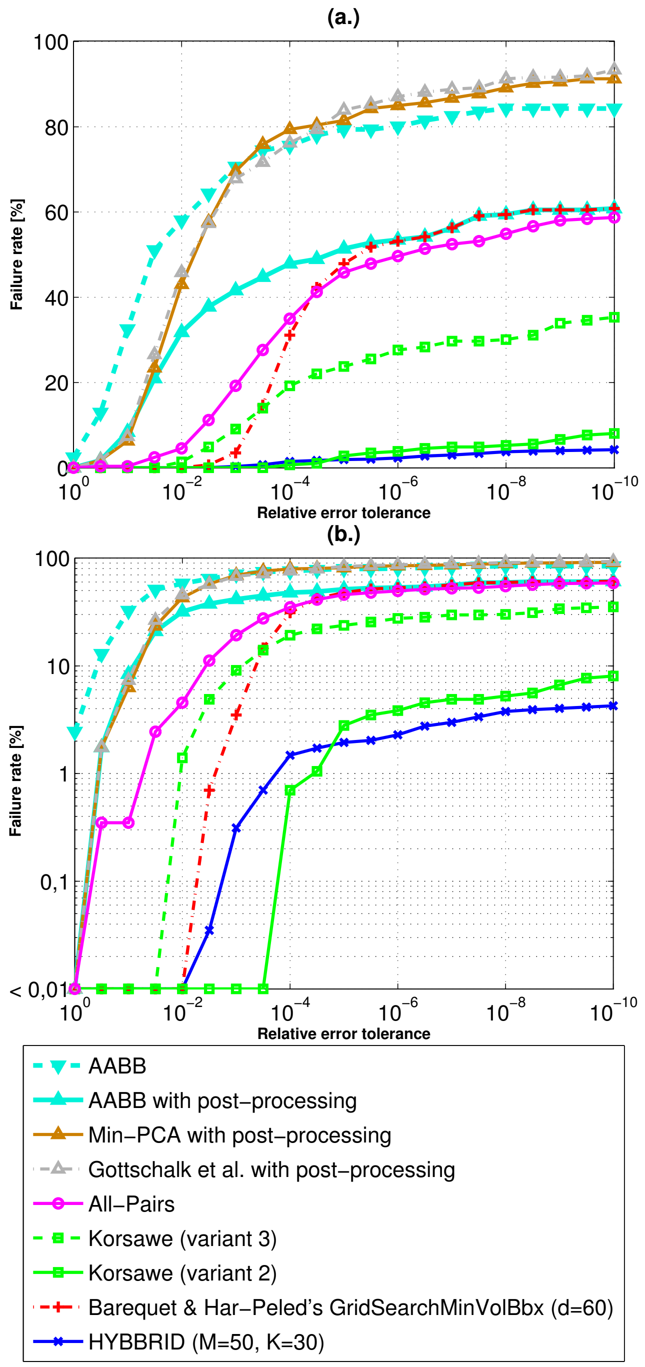


Fig. 13. Failure rate of the different algorithms for several tolerance thresholds. The same results are displayed with a linear vertical axis in (a.) and with a logarithmic vertical axis in (b.). Computations have been done on the whole set of about 300 objects. For each test case and each threshold τ, the run is considered as a failure if the relative error is greater than τ. For HYBBRID, the failure rate is based on 200 runs for each object.

than the result obtained by an axis-aligned bounding box. Note that All-PCA, Min-PCA and Max-PCA are both represented by four points. Indeed, the obtained OBB may vary depending on if one applies the preprocessing step and/or the post-processing step, or none of them. In the case of continuous PCA, the preprocessing step is required by the algorithm; hence, Gottschalk et al.'s method is only represented by two points.

It appears that HYBBRID is consistently the fastest method delivering the optimal volume on almost all test cases, with sufficiently large parameters. Faster methods present considerably lower performance levels. Some implementation details must also be taken into account while reviewing those results. Regarding the speed of the PCA-based methods, those only use a few matrix operations, most of which are written in highly efficient C code in Matlab® . On the other hand, the other algorithms such as HYBBRID and O'Rourke's mostly use Matlab® code, which is slower. A well-written pure C implementation of those methods should reduce the execution time, making them better suited for real-life scenarios.

Another point of interest is the distribution of the error compared to the optimal solution over all runs for all test cases. This information is shown in Fig. 13 for a selected subset of algorithms. To each method correspond at least one curve in the figure obtained with the variants yielding the best performance, independently of the computation time. Of course, O'Rourke's algorithm is not represented on this figure since it is optimal and thus, has a failure rate of 0 %. A closer view with a logarithmic scale is also included in order to emphasize the exact performance of HYBBRID.

Note that the information about the failure rate for relative error tolerances of 10^{-2}, 10^{-3} and 10^{-8} was already contained in Fig. 12. However, it is also interesting to see the evolution of the failure rate with the tolerance between and outside those values. For instance, this allows to show that a failure rate of 50 % can be achieved with Min-PCA for smaller tolerances than with AABB without post-processing. Nevertheless, the latter has a slightly smaller failure rate for tolerances tending to 0, even though both methods are clearly unsuitable for such tolerance levels. It also appears that AABB with post-processing has a failure rate equivalent to that of Min-PCA for relative error tolerances bigger than 0.05, but is 30 % more reliable for very small tolerances.

A similar trade-off between the performances for small and large tolerances is observed for Barequet & Har-Peled's `GridSearchMinVolBbx` method (which is shown in Fig. 13 with parameter $d = 60$) with respect to the third variant of Korsawe's method and all-pairs. For instance, `GridSearchMinVolBbx` is more reliable than Korsawe's method (resp. all-pairs) for relative error tolerances larger than about 3×10^{-4} (resp. 4×10^{-5}) but then, the failure rate increases and becomes worse for smaller tolerances. Indeed, the reliability of the method is then roughly equivalent to that of AABB with post-processing for tolerances smaller than 10^{-6}. Of course, the value of the tolerance where `GridSearchMinVolBbx` becomes equivalent to AABB with post-processing approaches 0 as d increases.

The behaviors of HYBBRID and the second variant of Korsawe's method are better shown with a logarithmic scale. The thresholding of the failure rate at 0.01 % is only due to the finite sampling issue, i.e., only about 300 test cases. It appears that HYBBRID is the most reliable method for tolerances smaller than 10^{-5} ; more precisely, its failure rate is about twice as small as the second variant of Korsawe's method. The fact that HYBBRID is slightly less reliable than the latter for larger tolerances can be nuanced by taking into account the computation time as Korsawe's method is about 10000 times slower than HYBBRID to complete the set of test cases.

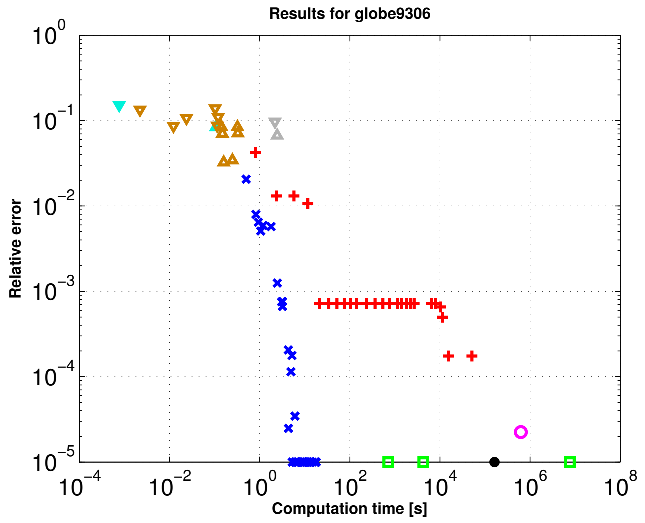


Fig. 14. Errors and computation time of all the algorithms on the example with the largest number of vertices on its convex hull. Computer specifications are identical to what is described for Table I. The computation time corresponds to the time required to approximate an optimal OBB, including the computation of the convex hulls if it is done by the algorithm. For HYBBRID, the volume of the OBB and computation time of each test case have been obtained by running the algorithm 200 times and averaging the results. The legend is the same as in Fig. 12. For the sake of readability, errors below 10^{-5} are displayed at this threshold value.

It is also interesting to compare the efficiency of all these methods on the most complex test case, i.e., `globe9306` in Fig. 7. The errors and computation times observed for this example are shown in Fig. 14. Among the pairs of parameters of HYBBRID represented on this figure, the one yielding the slowest computation in average is $(M, K) = (50, 30)$ with a running time of less than 18 seconds. Conversely, O'Rourke's algorithm needs about 45 hours to find the optimal OBB, and the three variants of Korsawe's method have a computation time ranging from 12 minutes to about 90 days.

5. CONCLUSION

In this article, HYBBRID, an algorithm to approximate the minimal-volume bounding box of a set of points, has been presented. It is a combination of two optimization components, namely the genetic algorithm and the Nelder-Mead algorithm. Combining those two methods leads to a method that shows a good convergence rate while still ensuring a good exploration of the search space in order to try to avoid local minima. The new idea in this article was to use such a hybrid method on the rotation group $SO(3, \mathbb{R})$ to determine the orientation corresponding to the smallest enclosing box, or OBB, of a set of points.

The new method has been compared to currently used algorithms, and has been shown to be either much faster than the fastest exact algorithm (O'Rourke's) or more accurate than the fastest heuristics based on principal component analysis. HYBBRID was able to find the optimal volume for each of the test cases we tried it on, which shows the reliability of the method. In order to test this algorithm, a broad set of methods was implemented in Matlab® . The whole codebase provides what we believe is a good framework to compare OBB fitting methods.

The scheme presented in this article could of course become the basis for further work. The same algorithm could be used to solve other problems on $SO(3, \mathbb{R})$, like for example, not finding the minimal volume OBB, but the one with the smallest area. It could also be interesting to investigate if such hybrid combinations of optimization methods could be used for the computations of other bounding volumes such as spheres or k-DOPs with arbitrary normals. Moreover, extensions to dimensions higher than 3 may be considered in areas such as data mining.

Another possible extension is the case with several distinct optimal OBBs. It may be useful to be able to find all the optimal solutions and select the best one according to another criterion. Since nothing in HYBBRID prevents the simplices from converging to different local minima, we only have to analyze the populations and detect all the global solutions that have been found. The genetic algorithm should then be modified to help the simplices converge to different global minima, e.g., by introducing a repulsive component. As there would be no guarantee that *all* solutions have been found, it would also be important to modify the parameters such as the population size in order to increase the probability of detecting all global minima.

ACKNOWLEDGMENTS

Chia-Tche Chang and Samuel Melchior are Research fellows with the Belgian National Fund for Scientific Research (FNRS). Bastien Gorissen is a first doca research fellow with the "Région wallonne" and Centre of Excellence in Aeronautical Research (Cenaero).

We would like to thank — by alphabetic order of last name — Pierre-Antoine Absil, Vincent Blondel, Pierre Borckmans, François Glineur, Koen Hillewaert, Vincent Legat, Jean-François Remacle, and Paul Van Dooren for their valuable comments.

We also express our thanks to the Associate Editor and the anonymous reviewers for their careful reading and valuable comments that helped us to improve the text.

REFERENCES

AGARWAL, P., HAR-PELED, S., AND VARADARAJAN, K. 2004. Approximating extent measures of points. *Journal of the ACM (JACM) 51,* 4, 606–635.

ASSARSSON, U. AND MÖLLER, T. 2000. Optimized view frustum culling algorithms for bounding boxes. *Journal of graphics, GPU, and game tools 5,* 1, 9–22.

BAREQUET, G., CHAZELLE, B., GUIBAS, L., MITCHELL, J., AND TAL, A. 1996. BOXTREE: A hierarchical representation for surfaces in 3D. In *Computer Graphics Forum*. Vol. 15:3. Wiley Online Library, 387–396.

BAREQUET, G. AND HAR-PELED, S. 2001. Efficiently Approximating the Minimum-Volume Bounding Box of a Point Set in Three Dimensions. *Journal of Algorithms 38,* 1, 91–109.

BORCKMANS, P. B. AND ABSIL, P.-A. 2010. Oriented bounding box computation using particle swarm optimization. In *ESANN 2010 proceedings, European Symposium on Artificial Neural Networks — Advances in Computational Intelligence and Learning*. D-side publications, Bruges, Belgium, 345–350.

CHAN, T. 2006. Faster core-set constructions and data-stream algorithms in fixed dimensions. *Computational Geometry: Theory and Applications 35,* 1-2, 20–35.

CHELOUAH, R. AND SIARRY, P. 2003. Genetic and nelder-mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research 148,* 2, 335–348.

CONN, A. R., GOULD, N. I. M., AND TOINT, P. L. 2000. *Trust Region Methods*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

CONN, A. R., SCHEINBERG, K., AND VICENTE, L. N. 2009. *Introduction to Derivative-Free Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

DEN BERGEN, G. V. 1998. Efficient collision detection of complex deformable models using AABB trees. *Journal of Graphics Tools* 2, 1–13.

DIMITROV, D., HOLST, M., KNAUER, C., AND KRIEGEL, K. 2009a. Closed-form solutions for continuous pca and bounding box algorithms. *Computer Vision and Computer Graphics. Theory and Applications 24*, 26–40.

DIMITROV, D., KNAUER, C., KRIEGEL, K., AND ROTE, G. 2009b. Bounds on the quality of the pca bounding boxes. *Computational Geometry: Theory and Applications 42,* 8, 772–789. Special Issue on the 23rd European Workshop on Computational Geometry.

DURAND, N. AND ALLIOT, J.-M. 1999. A combined nelder-mead simplex and genetic algorithm. In *GECCO'99: Proceedings of the 1999 Genetic and Evolutionary Computation Conference*. Morgan Kaufmann, Orlando, FL, USA, 1–7.

EDELSBRUNNER, H. AND MAURER, H. A. 1985. Finding extreme points in three dimensions and solving the post-office problem in the plane. *Information processing letters 21,* 1, 39–47.

ERICSON, C. 2004. *Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3-D Technology) (The Morgan Kaufmann Series in Interactive 3D Technology)*. Morgan Kaufmann.

FREEMAN, H. AND SHAPIRA, R. 1975. Determining the minimum-area encasing rectangle for an arbitrary closed curve. *Communications of the ACM 18,* 7, 409–413.

GAMMA GROUP. 2008. 3d meshes research database of the group Génération Automatique de Maillages et Méthodes d'Adaptation, INRIA, France. [Software, website]. Available at http://www-roc.inria.fr/gamma/gamma/download/download.php.

GARCIA-ALONSO, A., SERRANO, N., AND FLAQUER, J. 1994. Solving the collision detection problem. *IEEE Computer Graphics and Applications 14,* 3, 36–43.

GEUZAINE, C. AND REMACLE, J.-F. 2009. Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering 79*, 1309–1331.

GLOVER, F. 1989. Tabu Search — Part I. *ORSA Journal on Computing 1,* 3, 190–206.

GLOVER, F. 1990. Tabu Search — Part II. *ORSA Journal on Computing 2,* 1, 4–32.

GOLDBERG, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

GOLUB, G. H. AND VAN LOAN, C. F. 1996. *Matrix Computations*, 3rd ed. Johns Hopkins University Press.

GOTTSCHALK, S., LIN, M. C., AND MANOCHA, D. 1996. Obbtree: a hierarchical structure for rapid interference detection. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, New York, NY, USA, 171–180.

HOLLAND, J. H. 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI. second edition, 1992.

IONES, A., ZHUKOV, S., AND KRUPKIN, A. 1998. On optimality of obbs for visibility tests for frustum culling, ray shooting and collision detection. In *Computer Graphics International Conference*. IEEE Computer Society, Los Alamitos, CA, USA, 256.

JIMÉNEZ, P., THOMAS, F., AND TORRAS, C. 2000. 3d collision detection: A survey. *Computers and Graphics 25*, 269–285.

JOLLIFFE, I. T. 2002. *Principal Component Analysis*. Springer, New York, NY, USA.

KELLEY, C. T. 1999. *Iterative Methods for Optimization*. Number 18 in Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

KIRKPATRICK, S., GELATT, C. D., AND VECCHI, M. P. 1983. Optimization by simulated annealing. *Science 220,* 4598, 671–680.

KOLDA, T. G., LEWIS, R. M., AND TORCZON, V. 2003. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review 45,* 3, 385–482.

KORSAWE, J. 2008. Minimal bounding box. MATLAB Central File Exchange. [Software, MATLAB function]. Available at http://www.mathworks.com/matlabcentral/fileexchange/18264.

KRAKOWSKI, K. A., HÜPER, K., AND MANTON, J. H. 2007. On the computation of the karcher mean on spheres and special orthogonal groups. In *RoboMat 2007, Workshop on Robotics and Mathematics*. Coimbra, Portugal, 119–124.

LAHANAS, M., KEMMERER, T., MILICKOVIC, N., KAROUZAKIS, K., BALTAS, D., AND ZAMBOGLOU, N. 2000. Optimized bounding boxes for three-dimensional treatment planning in brachytherapy. *Medical Physics 27,* 10, 2333–2342.

MCKINNON, K. 1999. Convergence of the Nelder-Mead simplex method to a nonstationary point. *SIAM Journal on Optimization 9,* 1, 148–158.

NELDER, J. A. AND MEAD, R. 1965. A simplex method for function minimization. *The Computer Journal 7,* 4 (January), 308–313.

O'ROURKE, J. 1985. Finding minimal enclosing boxes. *International Journal of Computer & Information Sciences 14*, 183–199.

PONAMGI, M. K., MANOCHA, D., AND LIN, M. C. 1997. Incremental algorithms for collision detection between polygonal models. *IEEE Transactions on Visualization and Computer Graphics 3,* 1, 51–64.

PREPARATA, F. AND SHAMOS, M. 1985. *Computational geometry: an introduction*. Springer, New York, NY, USA.

PRESS, W., TEUKOLSKY, S., VETTERLING, W., AND FLANNERY, B. 1992. *Numerical Recipes in C*, 2nd ed. Cambridge University Press, Cambridge, UK.

REMACLE, J.-F., GEUZAINE, C., COMPÈRE, G., AND MARCHANDISE, E. 2010. High quality surface meshing using harmonic maps. *International Journal for Numerical Methods in Engineering 83*, 403–425.

SARLETTE, A. AND SEPULCHRE, R. 2009. Consensus optimization on manifolds. *SIAM Journal on Control and Optimization 48,* 1, 56–76.

SHAMOS, M. I. 1978. Computational geometry. Ph.D. thesis, Yale University, New Haven, CT, USA.

TOUSSAINT, G. 1983. Solving geometric problems with the rotating calipers. In *In Proceedings of IEEE MELECON '83*. Athens, Greece, A10.02:1–4.

Received December 2009; accepted April 2011