Reducing the time complexity and identifying ill-posed problem instances of Minkowski sum based similarity calculations

Bekker H. [1], Brink A. A.[2], Roerdink J.B.T.M.[1]

[1]Institute for Mathematics and Computing Science, [2]Dept. of Artificial Intelligence,
University of Groningen,
P.O.B. 800 9700 AV Groningen, The Netherlands.
bekker@cs.rug.nl, a.a.brink@ai.rug.nl, j.b.t.m.roerdink@rug.nl

Abstract. To calculate the Minkowski-sum based similarity measure of two convex polyhedra, many relative orientations have to be considered. These relative orientations are characterized by the fact that some faces and edges of the polyhedra are parallel. For every relative orientation of the polyhedra, the volume or mixed volume of their Minkowski sum is evaluated. From the minimum of this volume, the similarity measure is calculated. In this article two issues are addressed. First we propose and test a method to reduce the set of relative orientations to be considered by using geometric inequalities in the slope diagrams of the polyhedra. In this way the time complexity of $O(n^6)$ is reduced to $O(n^{4.5})$. Secondly we determine which relative orientation problems are ill-posed and may be skipped because they do not maximize the similarity measure.

1 Introduction

Shape comparison is of fundamental importance in computer vision. Therefore, in the past many families of methods to calculate the similarity of two shapes have been proposed. Well-known families are based on the Hausdorff metric, on contour descriptors and on moments of the object, see [1] for an overview. Some time ago a new family of methods has been introduced, based on the Minkowski inequality and its descendants. The central operation of this method is the minimization of a volume or mixed volume functional, over a set of relative orientations, of the Minkowski sum of two shapes that are to be compared [2,3,4]. It is defined for convex objects, and can be used to calculate many types of similarity measures. It is invariant under translation and rotation, and when desired, under scaling, reflection, isometries and similitudes [3]. In this article we limit ourselves to translation and rotation invariance. The method may be used in a space of any dimension, but we will concentrate on the 3D case. In [4] a specific application is presented.

To calculate the Minkowski sum based similarity of two convex sets in 3D space an infinite number of relative orientations has to be be considered. However, for general convex polyhedra only a finite number has to be considered, called the *critical* orientations. An orientation is called critical when three faces of one polyhedron are parallel with three edges of the other polyhedron. Thus, the set of critical orientations can be found by treating all combinations of three faces in one polyhedron and three edges in the other, and for every combination rotating one of the polyhedra such that the concerned combination of faces and edges becomes parallel. Methods to find such rotations, for three given edges and faces, are presented in [6].

In this article two main issues are addressed. The first issue is reducing the time complexity of this kind of similarity measure calculations. Experiments have previously been performed on 2D polygons [2] and 3D polyhedra [3,4], and show that for polygons the time consumption is low. However, even for 3D polyhedra of moderate complexity in terms of the number of faces, edges and vertices, the time consumption is prohibitive because a large number of relative orientations has to be considered. We present a method to reduce the time complexity for 3D polyhedra by reducing the number of relative orientations to be considered. The method determines, before critical orientations are calculated for three given faces and edges, whether it is possible that these become parallel simultaneously. In this way the number of critical orientations that actually has to be processed is reduced, resulting in a lower time complexity of the algorithm.

The second main issue that is addressed in this article, is ensuring robustness by detecting ill-posed critical orientation problems. For non-general convex polyhedra, the set of critical orientations is not always finite. That is, polyhedra with parallel and perpendicular edges and faces may result in ill-posed problems. Some ill-posed problem instances are mentioned in [3]. We derive a general classification scheme identifying all ill-posed problem instances. Using this scheme we show that the two classes of critical orientations in [3], called point-double and multiply-critical orientations, may be treated in a unified way. In [5] it was shown that critical orientations from both classes have to be considered. It has been shown in [3] that ill-posed problem instances do not minimize the volume or mixed volume functional, hence, they may be disregarded. The reason for identifying and disregarding ill-posed problem instances is that in this way it is avoided that singular or near-singular equations are solved, which would result in meaningless or erroneous results.

The structure of this article is as follows. In section 2 we introduce the Minkowski sum, the notion of mixed volume, the Brunn-Minkowski inequalities, and we derive some example similarity measures. Also the slope diagram representation is introduced and critical orientations are defined in terms of slope diagrams. In section 3 it is explained how geometric inequalities may be used to reduce the number of critical orientation calculations. In section 4 a classification scheme is derived, identifying ill-posed instances of the critical orientation calculation. In section 5 the implementation of the methods is discussed and in section 6 results from our experiments are presented. In section 7 a derivation of the observed time complexity is given.

2 Preliminaries

2.1 Minkowski sum based similarity measures

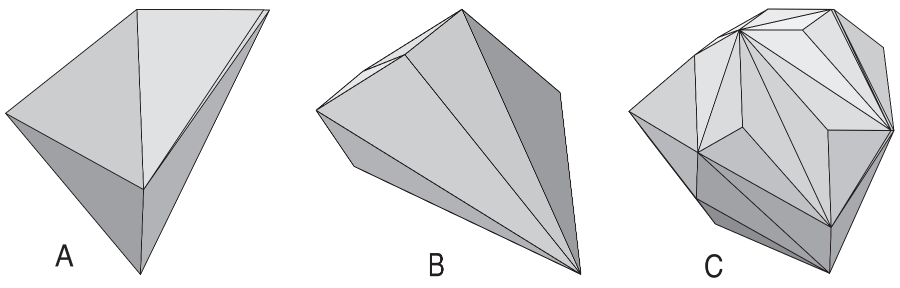

Fig. 1. Two polyhedra A and B and their Minkowski sum C. C is shown on half the scale of A and B.

The considered similarity measures are based on the Minkowski sum of two convex polyhedra. The Minkowski sum C of two sets A and B is defined as

$$C = A \oplus B \equiv \{a + b \mid a \in A, b \in B\}, \quad A, B, C \in \mathbb{R}^n. \tag{1}$$

In this article A and B are convex polyhedra in 3D space. Then C is also a convex polyhedron, generally with more faces, edges and vertices than A and B, see figure 1. Obviously, the shape and volume of C depend on the relative orientation of A and B. The volume of C may be written as

$$V(C) = V(A \oplus B) = V(A) + 3V(A, A, B) + 3V(A, B, B) + V(B). \tag{2}$$

Here, $V(A)$ and $V(B)$ are the volumes of A and B, and $V(A, A, B)$ and $V(A, B, B)$ are *mixed volumes*, introduced by Minkowski [8].

Many inequalities about volume and mixed volume are known, for example

1. The Minkowski inequality:
   For two arbitrary convex sets A and B in $\mathbb{R}^n$,

   $$V(A, A, B)^3 \geq V(A)^2 V(B) \tag{3}$$

   with equality if and only if $A = B$.
2. The Brunn-Minkowski inequality:
   For two arbitrary convex sets A and B in $\mathbb{R}^n$,

   $$V(A \oplus B) \geq 8V(A)^{\frac{1}{2}} V(B)^{\frac{1}{2}} \tag{4}$$

   with equality if and only if $A = B$.

From each of these inequalities a similarity measure may be derived [3],

$$\sigma(A, B) \equiv \max_{\mathbf{R} \in \mathcal{R}} \frac{V(A)^{2/3} V(B)^{1/3}}{V(\mathbf{R}(A), \mathbf{R}(A), B)} \tag{5}$$

$$\sigma_1(A, B) \equiv \max_{\mathbf{R} \in \mathcal{R}} \frac{8V(A)^{\frac{1}{2}} V(B)^{\frac{1}{2}}}{V(\mathbf{R}(A) \oplus B)}. \tag{6}$$

Here $\mathcal{R}$ denotes the set of all spatial rotations, and $\mathbf{R}(A)$ denotes a rotation of A by $\mathbf{R}$. Because the volumes in these equations are always positive, the similarity measures σ and σ_1 are always positive and ≤ 1, with equality if and only if $A = B$, up to translation. In [3] some other properties of σ and σ_1 are given.

Besides the Minkowski and the Brunn-Minkowski inequalities many other inequalities exist, some based on the volume, some on the mixed volume, some on the area and some on the mixed area of the Minkowski sum. From every of these inequalities a similarity measure may be derived. In this article we concentrate on computing σ. The technique presented in this article to speed up this computation may be applied to other Minkowski sum based similarity calculations as well.

To find the maximum in (5) for convex sets in general, an infinite number of orientations of A has to be considered. However, when A and B are convex polyhedra in 3D space, in general only a finite number of relative orientations of A and B has to be checked [3]. Roughly speaking these orientations are characterized by the fact that edges of B are as much as possible parallel to faces of A. This can be formulated in an exact way by using the slope diagram representation (SDR) of the polyhedra [7].

2.2 Slope diagram representation

In this article the slope diagram representation plays an important role as it is suitable to represent critical orientations. Critical orientations are defined in terms of relative orientations of slope diagrams. The slope diagram of a convex polyhedron in 3D space consists of points, spherical arcs of a great circle and spherical convex polygons on the unit sphere.

The SDR of a polyhedron A, denoted by SDR(A), is a subdivision on the unit sphere centered at the origin. A vertex of A is represented in SDR(A) by a spherical polygon, an edge by a spherical arc of a great circle, and a face by a vertex of a spherical polygon, see figure 2. Denoting face i of polyhedron A by $F_i(A)$, edge j by $E_j(A)$, and vertex k by $V_k(A)$ the SDR of a polyhedron is constructed as follows.

- *Face representation.* $F_i(A)$ is represented on the sphere by a point $SDR(F_i(A))$, located at the intersection of the outward unit normal vector u_i on $F_i(A)$ with the unit sphere.
- *Edge representation.* An edge $E_j(A)$ is represented by the arc of the great circle connecting the two points corresponding to the two adjacent faces of $E_j(A)$.

– *Vertex representation.* A vertex $V_k(A)$ is represented by the polygon bounded by the arcs corresponding to the edges of A meeting at $V_k(A)$.

Four remarks. SDR(A) is not a complete description of A, it only contains directional information about A. When A is rotated by a rotation $\mathbf{R}$, the slope diagram representation rotates in the same way, i.e., $SDR(\mathbf{R}(A)) = \mathbf{R}(SDR(A))$. In the following, when speaking about distance in an SDR we mean spherical distance, i.e., the length of the arc of a great circle on the unit sphere. Because the angle between two adjacent faces of a non-degenerate convex polyhedron is always less than π, the length of an arc in an SDR is always less than π.

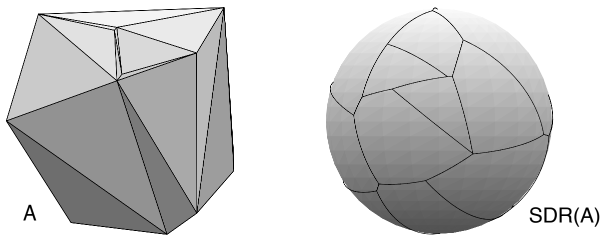

Fig. 2. A polyhedron A and SDR(A), i.e., the slope diagram representation of A. The orientations of A and $SDR(A)$ are the same, so it is possible to see how they are related.

2.3 Critical orientations

The slope diagram representations of two convex polyhedra are used to define their critical orientations. An orientation is called critical when, in the superimposed slope diagrams, three points of one slope diagram coincide with three arcs of the other slope diagram.

Two cases are distinguished.

case 1 Three spherical points of $SDR(\mathbf{R}(A))$ coincide with three spherical arcs of $SDR(B)$; See figure 3.

case 2 A spherical point of $SDR(\mathbf{R}(A))$ coincides with a spherical point of B, and simultaneously one spherical point of $SDR(\mathbf{R}(A))$ coincides with a spherical arc of $SDR(B)$.

Case 1 means that three edges of B are parallel to three faces of $\mathbf{R}(A)$. Case 2 means that a face of $\mathbf{R}(A)$ is parallel to a face of B, and an edge of B is parallel to a face of $\mathbf{R}(A)$. In [3] case 1 and case 2 are called multiply-critical and point-double respectively. Possibly, σ obtains its maximum when case 1 or case 2 fulfilled [3,4]. As will be shown later, case 2 is a special instance of case 1. Therefore, in the following we will only consider case 1.

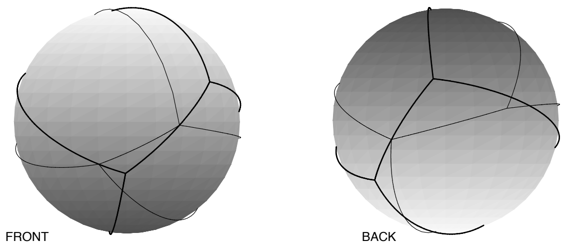

Fig. 3. Front side and back side of two superimposed slope diagrams. Thin lines: $SDR(\mathbf{R}(A))$. Bold lines: $SDR(B)$. At the front side two points of $SDR(\mathbf{R}(A))$ coincide with two arcs of $SDR(B)$, and at the back side one point of $SDR(\mathbf{R}(A))$ coincides with one arc of $SDR(B)$. So, in total three points of $SDR(\mathbf{R}(A))$ coincide with three arcs of $SDR(B)$, which corresponds with case 1.

3 Using geometric inequalities to skip unfeasible critical orientations

An orientation is called critical when in the superimposed slope diagrams three points of one slope diagram coincide with three arcs of the other slope diagram. In this section we show how some combinations of points and arcs can be skipped, resulting in speed-up.

Consider the unit sphere centered at the origin, with three arcs $\mathcal{K}, \mathcal{L}, \mathcal{M}$ and three points a, b, c; see figure 4. As explained before, a rotation $\mathbf{R}$ has to be calculated with the property that the point $\mathbf{R}(a)$ coincides with arc $\mathcal{K}$, $\mathbf{R}(b)$ coincides with arc $\mathcal{L}$ and $\mathbf{R}(c)$ coincides with arc $\mathcal{M}$. Without actually calculating $\mathbf{R}$ it is possible to detect situations where no such $\mathbf{R}$ exists. For this purpose spherical distances are used in the following way. Consider two spherical points a and b with a spherical distance $\mathrm{d}(a,\ b)$, and two arcs $\mathcal{K}$ and $\mathcal{L}$, where $\mathrm{dmin}(\mathcal{K},\ \mathcal{L})$ and $\mathrm{dmax}(\mathcal{K},\ \mathcal{L})$ are the minimal and maximal distance between the arcs. Here, $\mathrm{dmin}(\mathcal{K},\ \mathcal{L})$ is defined as the minimum distance of the points q_1 and q_2 where q_1 is on on $\mathcal{K}$ and q_2 is on $\mathcal{L}$, i.e., $\mathrm{dmin}(\mathcal{K}, \mathcal{L}) \equiv min(d(q_1, q_2))$ where $q_1 \in \mathcal{K}$ and $q_2 \in \mathcal{L}$. The maximum distance $\mathrm{dmax}(\mathcal{K},\ \mathcal{L})$ is defined analogously. Obviously, only when $\mathrm{dmin}(\mathcal{K},\ \mathcal{L}) \leq \mathrm{d}(a, b) \leq \mathrm{dmax}(\mathcal{K},\ \mathcal{L})$, a can coincide with $\mathcal{K}$ while at the same time b coincides with $\mathcal{L}$, see figure 4. The same principle may be used for the other two pairs of points and arcs, i.e, a critical orientation calculation should only be performed when

$$\begin{aligned} \mathrm{dmin}(\mathcal{K}, \mathcal{L}) \leq \mathrm{d(a,b)} \leq \mathrm{dmax}(\mathcal{K}, \mathcal{L}) \ \text{ and} \\ \mathrm{dmin}(\mathcal{L}, \mathcal{M}) \leq \mathrm{d(b,c)} \leq \mathrm{dmax}(\mathcal{L}, \mathcal{M}) \ \text{ and} \\ \mathrm{dmin}(\mathcal{M}, \mathcal{K}) \leq \mathrm{d(c,a)} \leq \mathrm{dmax}(\mathcal{M}, \mathcal{K}). \end{aligned} \tag{7}$$

Problem instances that do not fulfil (7) are unfeasible, the opposite is not necessarily true. That is because in (7) only distances are considered, and not, for example, angles.

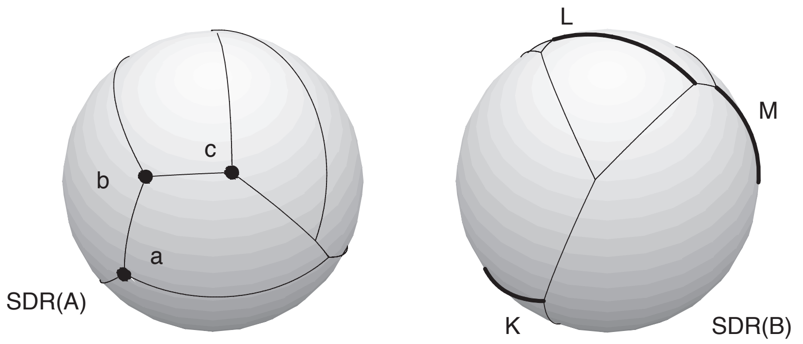

Fig. 4. $SDR(A)$ with three marked points a, b, c and $SDR(B)$ with three marked arcs $\mathcal{K}$, $\mathcal{L}$, $\mathcal{M}$. $SDR(A)$ may be rotated so that in the overlay $\mathbf{R}(b)$ coincides with $\mathcal{L}$ and $\mathbf{R}(c)$ coincides with $\mathcal{M}$, but clearly then $\mathbf{R}(a)$ can not coincide with $\mathcal{K}$. I.e., the situation that $\mathbf{R}(b)$ coincides with $\mathcal{L}$, $\mathbf{R}(c)$ coincides with $\mathcal{M}$ and $\mathbf{R}(a)$ coincides with $\mathcal{K}$ is unfeasible.

4 Identifying ill-posed problem instances

No method is known that calculates a rotation with the property formulated in case 1 in a direct way. Instead, a two-step approach is used for this. First an orientation is calculated where three points coincide with three great circles carrying the given arcs. Secondly, it is checked whether the three rotated points coincide with the three arcs, which is a trivial calculation. In this section ill-posed problem instances will be considered, as encountered in the first step.

Consider the unit sphere centered at the origin, with three great circles k, l, m, carrying three arcs $\mathcal{K}, \mathcal{L}, \mathcal{M}$, each representing an edge of polyhedron B. The vectors $\mathbf{k}, \mathbf{l}, \mathbf{m}$ are normals of the planes through the origin, carrying the great circles, i.e., they are the normals of the great circles. Whether the normals are on the positive or negative side of the planes is of no importance. On the unit sphere are also given three points a, b, c. These points are the end-points of the unit vectors $\hat{\mathbf{a}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}$, where $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are the normals of faces of A. The lengths of $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{k}, \mathbf{l}, \mathbf{m}$ are finite and non-zero, and play no role in the critical orientation calculations.

To calculate critical orientations it is required to calculate the proper orthogonal transformation $\mathbf{R}$ with the property

$$\mathbf{k} \cdot \mathbf{R}(\mathbf{a}) = 0, \quad \mathbf{l} \cdot \mathbf{R}(\mathbf{b}) = 0, \quad \mathbf{m} \cdot \mathbf{R}(\mathbf{c}) = 0 \tag{8}$$

where $\cdot$ denotes the standard inner product. I.e., after rotating over $\mathbf{R}$ the points a, b, c coincide with the great circles k, l, m respectively. The matrix $mat(\mathbf{R})$, which is the matrix representation of $\mathbf{R}$, represents a *proper orthogonal transformation* in 3D when
$\Sigma_i \, R_{ij} R_{ik} = \delta_{jk}, \;\; j, k = 1, 2, 3$ and $\det(mat(\mathbf{R})) = 1$.

How $\mathbf{R}$ may be calculated is discussed in [6]. There, only well-posed problem instances are considered, where *(ill-) well-* posed means that the number of solutions of (8) is (in)finite. The calculation is performed by writing (8) as a set of three second degree polynomial equations. It has been shown there that the number of solutions of a well-posed problem is $0, 2, 4, 6$ or 8. An example of an ill-posed problem instance is given by

$$\mathbf{k} \cdot \mathbf{R}(\mathbf{a}) = 0, \quad \mathbf{l} \cdot \mathbf{R}(\mathbf{a}) = 0, \quad \mathbf{m} \cdot \mathbf{R}(\mathbf{a}) = 0, \tag{9}$$

with $\mathbf{k}, \mathbf{l}, \mathbf{m}$ coplanar. I.e., the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ from (8) coincide. Clearly, any $\mathbf{R}$ that rotates $\mathbf{a}$ such that $\mathbf{R}(\mathbf{a})$ is perpendicular to the plane that carries $\mathbf{k}, \mathbf{l}, \mathbf{m}$ is a solution. However, any rotation $\mathbf{R}'(\mathbf{R}(\mathbf{a}))$, where $\mathbf{R}'$ is a rotation around $\mathbf{R}(\mathbf{a})$, is also a solution. There is an infinite number of rotations $\mathbf{R}'$ so the problem is ill-posed. When the vectors $\mathbf{k}, \mathbf{l}, \mathbf{m}$ are linearly independent the problem is well-posed and the number of solutions to (9) is zero. The goal of this section is to identify ill-posed problem types in a systematic way in order to skip them.

Equation (8) is said to be *degenerate* when the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are not in general relative position but coincide or are perpendicular, or when the same holds for $\mathbf{k}, \mathbf{l}, \mathbf{m}$. As will be clear from the foregoing, some degenerate instances of (8) are ill-posed and some well-posed. When two or three equations in (8) are identical we are obviously dealing with an ill-posed problem instance. In the following we assume that this type of ill-posedness has been filtered out already by a combinatorial check on the pointers to the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ and $\mathbf{k}, \mathbf{l}, \mathbf{m}$. Later, also a geometrical check will be performed to see whether the vectors are parallel or perpendicular.

Let us now actually determine which types of critical orientation problems are ill-posed. Suppose we have created in some way a situation where

$$\mathbf{k} \cdot \mathbf{a} = 0, \;\; \mathbf{l} \cdot \mathbf{b} = 0, \;\; \mathbf{m} \cdot \mathbf{c} = 0. \tag{10}$$

It is well-known that any proper orthogonal transformation $\mathbf{R}$ may be written as a rotation around an axis ρ over some angle. So an ill-posed problem instance is characterized by the fact that there is an infinite number of axes and rotation angles, each represented by its own $\mathbf{R}$, such that

$$\mathbf{k} \cdot \mathbf{R}(\mathbf{a}) = 0, \;\; \mathbf{l} \cdot \mathbf{R}(\mathbf{b}) = 0, \;\; \mathbf{m} \cdot \mathbf{R}(\mathbf{c}) = 0. \tag{11}$$

Our approach to derive a classification scheme of ill-posed problem types is to separately identify ill-posed instances for the pairs $(\mathbf{a}, \mathbf{k})$, $(\mathbf{b}, \mathbf{l})$ and $(\mathbf{c}, \mathbf{m})$. By combining ill-posed pair configurations we identify ill-posed configurations for the whole problem.

Assuming that $\mathbf{k} \cdot \mathbf{a} = 0$, there are two axes ρ, about which $\mathbf{a}$ may be rotated, giving $\mathbf{k} \cdot \mathbf{R}(\mathbf{a}) = 0$. One axis coincides with $\mathbf{a}$, the other with $\mathbf{k}$, see figure 5. These cases are denoted as 1 and 2. For the pairs ($\mathbf{l}$,$\mathbf{b}$) and ($\mathbf{m}$,$\mathbf{c}$) similar axes exist, see figure 5. These cases are also also denoted by 1 and 2.

Now we combine three ill-posed configurations, one from each pair, such that their three axes ρ coincide. For example, in the ill-posed configuration 1,1,1 the vectors $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$ coincide with each other and with ρ, while $\mathbf{k}$, $\mathbf{l}$, $\mathbf{m}$ are perpendicular to ρ, i.e., are coplanar, see figure 6. This configuration was discussed in the example above. In total there are eight different combinations, hence there are eight types of ill-posed problems. We use the following notation, where parallel means parallel or anti-parallel. When three vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are parallel this is represented by $\mathbf{a} \| \mathbf{b} \| \mathbf{c}$. When the vectors $\mathbf{a}$ and $\mathbf{b}$ are parallel and they are both perpendicular to $\mathbf{c}$ this is represented by $(\mathbf{a} \parallel \mathbf{b}) \perp \mathbf{c}$. When $\mathbf{b}$ is perpendicular to $\mathbf{a}$, and $\mathbf{c}$ is perpendicular to $\mathbf{a}$ this is represented by $\mathbf{b} \perp \mathbf{a} \wedge \mathbf{c} \perp \mathbf{a}$. When $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are coplanar this represented by $\mathit{copl}(\mathbf{a}, \mathbf{b}, \mathbf{c})$. Using this notation gives the scheme as shown in table 1. The index, for example $1, 1, 1$, tells which three situations have been combined. The right column describes the geometry of the configuration.

1,1,1	$\mathbf{a} \| \mathbf{b} \| \mathbf{c}$ and $\mathit{copl}(\mathbf{k}, \mathbf{l}, \mathbf{m})$
1,1,2	$(\mathbf{a} \| \mathbf{b}) \perp \mathbf{c}$ and $\mathbf{k} \perp \mathbf{m} \wedge \mathbf{l} \perp \mathbf{m}$
1,2,1	$(\mathbf{a} \| \mathbf{c}) \perp \mathbf{b}$ and $\mathbf{k} \perp \mathbf{l} \wedge \mathbf{m} \perp \mathbf{l}$
1,2,2	$(\mathbf{b} \| \mathbf{c}) \perp \mathbf{a}$ and $\mathbf{l} \perp \mathbf{k} \wedge \mathbf{m} \perp \mathbf{k}$
2,1,1	$\mathbf{b} \perp \mathbf{a} \wedge \mathbf{c} \perp \mathbf{a}$ and $(\mathbf{l} \| \mathbf{m}) \perp \mathbf{k}$
2,1,2	$\mathbf{a} \perp \mathbf{b} \wedge \mathbf{c} \perp \mathbf{b}$ and $(\mathbf{k} \| \mathbf{m}) \perp \mathbf{l}$
2,2,1	$\mathbf{a} \perp \mathbf{c} \wedge \mathbf{b} \perp \mathbf{c}$ and $(\mathbf{k} \| \mathbf{l}) \perp \mathbf{m}$
2,2,2	$\mathit{copl}(\mathbf{a}, \mathbf{b}, \mathbf{c})$ and $\mathbf{k} \| \mathbf{l} \| \mathbf{m}$

Table 1. Ill-posed problem types listed systematically. The first column gives combinations of three pairs, as shown in figure 5. In the second column the geometry of the configuration is given, i.e, whether vectors are parallel, perpendicular or coplanar.

Checking a problem instance for ill-posedness means that it has to be checked for the eight situations in this list, thus eight types of checks would have to be implemented. This number can be reduced to two by inverting and permuting the problem. In a problem *inversion* the vectors $\mathbf{a}$ and $\mathbf{k}$ are exchanged, as well as the vector $\mathbf{b}$ and $\mathbf{l}$, and $\mathbf{c}$ and $\mathbf{m}$. It can easily be seen that, when a problem is ill-posed, its inverse is also ill-posed. In a problem *permutation* the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are permuted in some way, and the vectors $\mathbf{k}, \mathbf{l}, \mathbf{m}$ are permuted similarly. Clearly, when a problem is ill-posed its permutations are also ill-posed. From table (1) it can be seen that, apart from permutations and inversions, there are actually only two ill-posed problem types. Case $[1, 1, 1]$ is the inverse of case $[2, 2, 2]$. The cases $[1, 1, 2], [1, 2, 1]$ and $[2, 1, 1]$ are identical, apart from a permutation. Similarly, the cases $[1, 2, 2], [2, 1, 2]$ and $[2, 2, 1]$ are identical, apart from a permutation. Finally, the cases $[1, 1, 2], [1, 2, 1]$ and $[2, 1, 1]$ are inverse cases of $[2, 2, 1], [2, 1, 2]$ and $[1, 2, 2]$ respectively. Thus only two classes of ill-posed problem types remain, the class consisting of $[1, 1, 1]$ and $[2, 2, 2]$, and the class consisting of $[1, 1, 2], [1, 2, 1]$, $[2, 1, 1]$, $[1, 2, 2], [2, 1, 2]$ and $[1, 2, 2]$. In this way only two types of checks have to be implemented and these are used to test permuted and inverted instances of the problem at hand.

From the analysis above it is clear that many degenerate problem instances are ill-posed. However, there are also many degenerate instances of (8) that

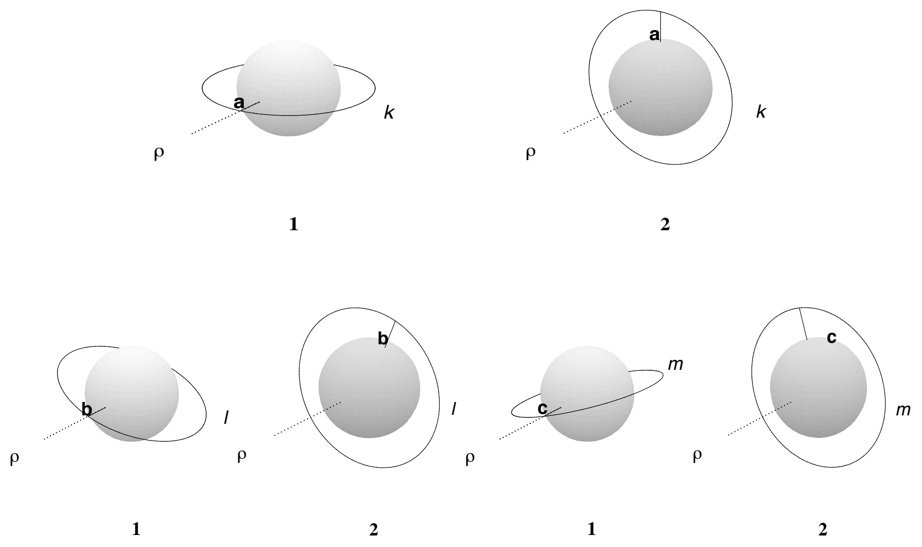

Fig. 5. Ill-posed problem types derived systematically. Six ill-posed configurations of the three pairs $(\mathbf{a}, \mathbf{k})$, $(\mathbf{b}, \mathbf{l})$ and $(\mathbf{c}, \mathbf{m})$ respectively, two for each pair. $\mathbf{k}, \mathbf{l}$ and $\mathbf{m}$ are not shown, but their great circles k, l and m. To improve visibility, instead of the unit sphere a sphere with radius 0.5 is shown.

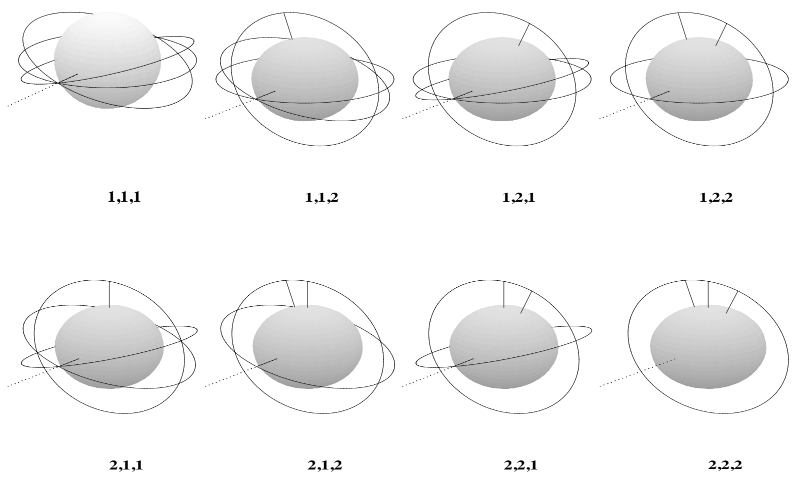

Fig. 6. All eight ill-posed configurations, as listed in table 1. Every figure is a combination of three figures from figure 5, one from each pair.

are well-posed. For example, see figure 7. The problem instance where $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are coplanar and $\mathbf{k}, \mathbf{l}, \mathbf{m}$ are coplanar is well-posed. Also the configuration where two of the points a, b, c coincide is well-posed. Case 2, i.e., the point-double situation is an instance of this type of configuration. See figure 7.

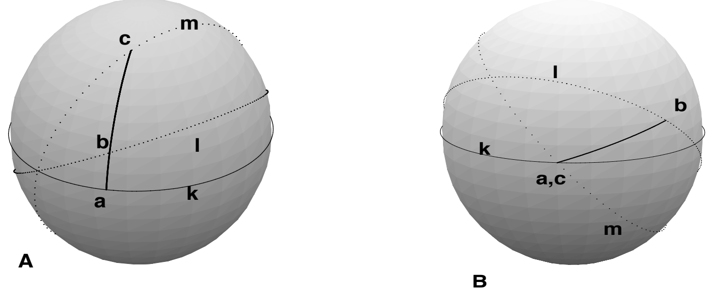

Fig. 7. Two degenerate problem instances that are well-posed. A: the vectors $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are coplanar and the vectors $\mathbf{k}, \mathbf{l}, \mathbf{m}$ are coplanar. For the visual effect the points a, b, c are connected by a line segment. B: Case 2, i.e., the point-double configuration. The points a and c coincide, so, when rotated they will coincide with one of the two intersection points of the great circles k and m. In both figures one of four solutions is shown.

5 Implementation

In the implementation an important role is played by a list L, which contains all pairs (p, a), where p is a point of SDR(A) and a is an arc of SDR(B). For general polyhedra it holds that the number of vertices, edges and faces are proportional to each other, so, in the slope diagram it holds that the number of points, arcs and faces are proportional to each other. Defining the *complexity* of a polyhedron as the number of faces, and assuming that A and B have the same complexity, the length of L is proportional to the squared complexity of A or of B.

Three more data structures are *d_pts*, *mind_arcs* and *maxd_arcs*. In *d_pts* the distance between every pair of points of SDR(A) is stored. In *mind_arcs (maxd_arcs)* the minimum (maximum) distance between every pair of arcs of SDR(B) is stored. The data structures are filled before the actual calculations start.

In the Boolean function *feasible_distances(i,j,k,L)* the calculations from (7) are implemented. Here, i, j, k are indices in L, so, the triple i, j, k defines three points and three arcs. The function *feasible_distances(i,j,k,L)* uses data in *d_pts, mind_arcs and maxd_arcs*. In the Boolean function *well_posed(i,j,k,L)* the tests from table 1 are implemented. The actual critical orientation calculations are implemented in the function *crit_or(i,j,k,L)*, see [6]. After calling this function the list R contains $0, 2, 4, 6$ or 8 rotation matrices. These rotation matrices are assigned to r successively. In the Boolean function *pts_on_arcs(i,j,k,L,r)* it is tested whether the three points defined by i, j, k, after rotating them over r, coincide with the three arcs defined by i, j, k. The result of these four functions is invariant under permutation of the arguments i, j, k, so a factor of six may be gained by iterating through i, j, k such that $i < j < k$. The similarity measure of A and B, as given in (5), is calculated in *σ(A,B,r)*, and the maximum value is stored. The complete algorithm is given in Algorithm 1.

Algorithm 1 New algorithm

```
INPUT: $A, B$ {convex polyhedra}
OUTPUT: $s$ {the similarity measure of A and B}
$SDR\_A \leftarrow construct\_slope\_diagram(A)$
$SDR\_B \leftarrow construct\_slope\_diagram(B)$
$d\_pts \leftarrow compute\_point\_distances($points in $SDR\_A)$
$mind\_arcs, maxd\_arcs \leftarrow compute\_arc\_distances($arcs in $SDR\_B)$
$L \leftarrow make\_combinations($points in $SDR\_A$, arcs in $SDR\_B)$
$n \leftarrow length(L)$
$s \leftarrow 0.0$
for all $i$ such that $0 \leq i < n - 2$ do
  for all $j$ such that $i + 1 \leq j < n - 1$ do
    for all $k$ such that $j + 1 \leq k < n$ do
      if $feasible\_distances(i, j, k, L)$ then
        if $well\_posed(i, j, k, L)$ then
          $R \leftarrow crit\_or(i, j, k, L)$ {R is a list of rotation matrices}
          for all $r$ in $R$ do {r is a rotation matrix}
            if $points\_on\_arcs(i, j, k, L, r)$ then
              $s \leftarrow \max(\sigma(A, B, r), s)$
            end if
          end for
        end if
      end if
    end for
  end for
end for
print ``The similarity measure of A and B is: '', s
```

In the following we refer to this algorithm as the *new algorithm*. This algorithm without the calls to *feasible_distances(i,j,k,L)* and *well_posed(i,j,k,L)* will be referred to as the *old algorithm*.

6 Results

6.1 Using geometric inequalities to skip unfeasible critical orientations

In this subsection we present the measured times of the old and new algorithm. To compare the speed of the old and the new algorithm we count for both

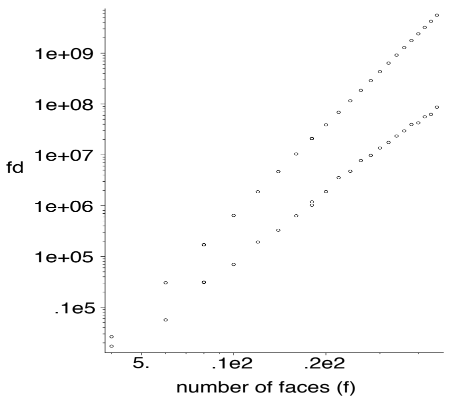

Fig. 8. The results of the experiments with the old algorithm (upper dots) and new algorithm (lower dots), plotted logarithmically on both axes. f is the number of faces of A and B. fd is the number of times the function *crit_or(i,j,k,L)* is called. The results of both algorithms are linear, indicating that the time complexity is of the form $fd = a.f^e$, where e is the exponent and a a constant. A least squares fit in this plot gives $e = 6.00$ for the old algorithm, and $e = 4.57$ for the new algorithm. So, the experimental time complexity of the old algorithm is $O(f^{6.00})$ and of the new algorithm $O(f^{4.57})$.

algorithms the number of times fd the function *crit_or(i,j,k,L)* is called. Obviously, the number of calls in the new algorithm will be less than the number of calls in the old algorithm. As the complexity of B and A increases, the arcs in $SDR(B)$ get smaller. The smaller the arcs, the smaller the range of distances between them and thus the smaller the probability that a pair of points will fit between them, resulting in a higher probability that combinations are skipped. I.e., we may expect a speed improvement that is not simply a constant factor but that is stronger for more complex polyhedra. The difference between fd for the old and the new algorithm is practically completely caused by the function *feasible_distances(i,j,k,L)*. As explained earlier, the effect of the function *well_posed(i,j,k,L)* on the complexity is negligible.

We tested the new and the old algorithm on randomly generated polyhedra, ranging in complexity from 4 to 46 faces. The polyhedra A and B had the same number of faces. For every test we generated a random polyhedron A, and generated random polyhedra B until a polyhedron was found with the same number of faces as A. For the pair A, B we used the new and the old algorithm to determine fd. In figure 8 the logarithm of fd is plotted as a function of the logarithm of the number of faces. From this plot it can be seen that the new algorithm is significantly faster than the old algorithm. For polyhedra with 10 faces the new algorithm is ≈ 10 times faster than the old algorithm, for polyhedra with 46 faces it is ≈ 60 times faster. In the log-log plot, the results of the new algorithm and the old algorithm are both linear, indicating that both algorithms have a time complexity of the form $fd = a.f^e$, where f is the number faces of

A and B, and a and e are constants. Fitting a line through these points with a least squares method gives for the old algorithm $e = 6.00$, and for the new algorithm $e = 4.57$. So, the experimental time complexity of the old algorithm is $O(f^{6.00})$ and of the new algorithm $O(f^{4.57})$. The $O(f^{6.00})$ time complexity of the old algorithm is obvious, considering that the length of the list L is proportional to f^2 and that it is processed in a threefold loop.

6.2 Identifying ill-posed problem instances

The most important effect of testing on ill-posedness is that the critical orientation calculations become robust. This was tested on various ill-posed problems. As ill-posed problems occur in particular when working with polyhedra with parallel and/or perpendicular edges and faces, we calculated amongst others the similarity measure of the five platonic solids with respect to each other and with respect to random polyhedra. Calculating the similarity of these shapes, without testing for ill-posedness, resulted for more than 50% of the similarity calculations of A and B in at least one ill-posed critical orientation calculation. After filtering out ill-posed critical orientation calculations, the calculations could be performed with the generic method from [6] and resulted in a finite number of solutions. I.e., testing for ill-posedness is an effective approach to make the critical orientation calculations robust.

7 Derivation of the observed time complexity

The time complexity of the new algorithm is mainly influenced by using geometric inequalities to skip unfeasible critical orientations. Although sometimes some critical orientation calculations are skipped as a result of testing on ill-posedness, as explained before, this will have negligible effect on the time complexity. Therefore, this section concentrates on using geometric inequalities to skip unfeasible critical orientations.

A simple derivation of the complexity of the new algorithm is as follows. The number of faces f, edges and vertices of a polyhedron are proportional to each other, so the number of faces, arcs and points in a slope diagram are also proportional to each other and proportional to f. Therefore, in the slope diagram the average area per arc is proportional to f^{-1}. The average area of a spherical polygon in the slope diagram is proportional to f^{-1}, hence the average length of an arc in the slope diagram is proportional to $f^{-1/2}$. Let us now consider the following situation. On the unit sphere we draw an arc a with length $|a|$, and we draw the region consisting of all points with a distance d to at least one point of a. See figure 9. The region consists of a belt with an average area proportional to $d|a| \propto df^{-1/2}$. The average number of arcs falling completely or partially in this belt is $\frac{\text{average belt area}}{\text{average arc area}}$, i.e this is proportional to $df^{1/2}$. Now we do not only consider the arc a but all f arcs. Hence, in a slope diagram the number of pairs of arcs $a1, a2$ for which it holds that $dmin(a1, a2) \leq d \leq dmax(a1, a2)$ is proportional to $df^{1.5}$. In (7) we are not considering one but three pairs of arcs, hence, the number times that for a given d (7) is fulfilled is proportional to $(df^{1.5})^3 = d^3 f^{4.5}$. The distance between two points in a slope diagram may range from 0 to π, and d does not depend on f. Therefore, the number times that (7) is fulfilled is proportional to $f^{4.5}$.

This agrees reasonably well with our experimental result of $f^{4.57}$. That our experimental result differs from the theoretical result seems to be caused by the fact that in our experiments we also used some polyhedra with few faces. In figure 8 it can be seen that the slope of the curve of the new algorithm decreases slightly for more complex polyhedra. Disregarding the first ten data points and fitting a line to the remaining points gives an exponent of 4.52. So, for more complex polyhedra our experimental complexity corresponds well with the derived complexity of $f^{4.5}$.

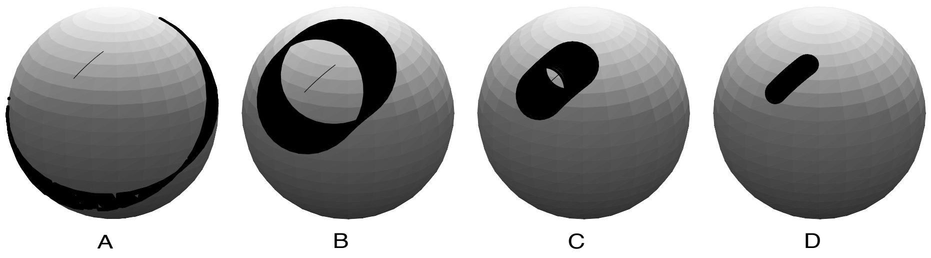

Fig. 9. Four spheres A, B, C, D, with an arc a, and regions (black) consisting of all points with a spherical distance of 1.5, 0.58, 0.25, 0.1 respectively, to at least one point of a. In C and D the arc is (partially) covered by the black area.

8 Discussion, conclusion and future work

In section 4 we showed that the case 2 type of critical orientation is a special instance of the case 1 type. As a result, case 2 can be handled by the critical orientation calculation method used for case 1. But potentially, in this way the calculations may become inefficient for the following reason. Case 2 means that, as with all critical orientations, three points a, b, c have to coincide with three arcs $\mathcal{K}, \mathcal{L}, \mathcal{M}$ respectively. Additionally, case 2 means that two points, say a and b, coincide. Obviously, then a critical orientation is only possible when $\mathcal{K}$ and $\mathcal{L}$ have a point in common. In a slope diagram, in general, arcs do not have points in common, only some arcs have coinciding end points. Thus, calculating a rotation when a and b coincide, and $\mathcal{K}$ and $\mathcal{L}$ have no point in common is a waste. Fortunately, this situation is signaled by the procedure *feasible_distances()*. I.e., the potential inefficiency, caused by unifying case 1 and case 2, is prevented by combining this unified approach with using the procedure *feasible_distances()*.

By comparing the distance between two points in a slope diagram with the range of distances between two arcs in a slope diagram a significant speedup of the critical orientation calculations is achieved. However, the time complexity is still $O(f^{4.57})$. We consider reducing this complexity further in two ways.

In the first place, instead of comparing distances we could compare angles in the following way. Let the angle defined by the points a, b, c be the angle between the two line segments a, b and b, c. Similarly, three points d, e, f, located on $\mathcal{K}$, $\mathcal{L}$ and $\mathcal{M}$ respectively, define a range of angles. When the angle defined by a, b, c does not fall into the range of angles defined by d, e, f there is no critical orientation for the points a, b, c and the arcs $\mathcal{K}$, $\mathcal{L}$, $\mathcal{M}$. Analogously, this criterion may be used for the other two angles defined by the points a, b, c and d, e, f.

A second possible approach is as follows. When point a coincides with $\mathcal{K}$ and point b coincides with $\mathcal{L}$ then the position of point c is not completely determined. The points a and b may move on $\mathcal{K}$ and $\mathcal{L}$ respectively, which causes point c to move on one or two segments of some curve. When these curve segments do not intersect arc $\mathcal{M}$ there is no critical orientation for this combination of points and arcs.

In this article we showed that, by applying simple geometric techniques, critical orientation calculations can be strongly reduced, without introducing significant computational overhead. We expect that, by combining the method proposed in this article with other simple geometric tests, the number of critical orientation calculations can be further reduced.

Literature

[1] Veltkamp, R.C. Shape Matching: Similarity Measures and Algorithms. *Shape Modeling International (2001), 188-196*

[2] Heijmans, H. J. A. M., Tuzikov, A. Similarity and symmetry measures for convex shapes using Minkowski addition. *IEEE Trans. Patt. Anal. Mach. Intell. 20*, 9 (1998), 980-993.

[3] Tuzikov, A. V., Roerdink, J. B. T. M., and Heijmans, H. J. A. M. Similarity measures for convex polyhedra based on Minkowski addition. *Pattern Recognition 33*, 6 (2000), 979-995.

[4] Tuzikov A. V., Sheynin S. A. Symmetry Measure Computation for Convex Polyhedra. *Journal of Mathematical Imaging and Vision* 16 (2002), 41–56.

[5] Roerdink J.B.T.M., Bekker H. Similarity measure computation of convex polyhedra revisited. *LNCS vol. 2243, (2001), 389-399 Springer Verlag.*

[6] Bekker, H., and Roerdink, J. B. T. M. Calculating critical orientations of polyhedra for similarity measure evaluation. In *Proc. 2nd Annual IASTED International Conference on Computer Graphics and Imaging, Palm Springs, California USA, Oct. 25-27* (1999), 106–111.

[7] Ghosh P. K., Haralick R. M. Mathematical morphological operations of boundary-represented geometric objects. *Journal of Mathematical Imaging and Vision* 6 (1996), 199-222

[8] Sangwine-Yager, J.R. Mixed volumes. Chapter 1.2 of: *Handbook of convex geometry.* (1993), Eds. Gruber, P.M., Wills, J.M. Elsevier Science Publishers B.V.