Contributing Vertices-based Minkowski sum of a non-convex polyhedron without fold and a convex polyhedron

Hichem Barki[1], Florence Denis[1], Florent Dupont[1]

[1]Université de Lyon, CNRS - Université Lyon 1, LIRIS, UMR5205 - 43 Bd. du 11 novembre 1918, F-69622 Villeurbanne, France

***Abstract*—We present an original approach for the computation of the Minkowski sum of a non-convex polyhedron without fold and a convex polyhedron, without decomposition and union steps—that constitute the bottleneck of convex decomposition-based algorithms. A non-convex polyhedron without fold is a polyhedron whose boundary is completely recoverable from three orthographic projections defined by three orthogonal basis vectors in $\mathbb{R}^3$. First, we generate a superset of the Minkowski sum facets using the concept of contributing vertices we accommodate for a non-convex–convex pair of polyhedra. The generated superset guarantees that its envelope is the boundary of the Minkowski sum polyhedron. Secondly, we extract the Minkowski sum facets and handle the intersections among the superset facets by using 3D envelope computation. Our approach is limited to non-convex polyhedra without fold because of the use of 3D envelope computation to recover the Minkowski sum boundary. Models with holes are not handled by our method. The implementation of our algorithm uses exact number types, produces exact results, and is based on CGAL, the Computational Geometry Algorithms Library.**

***Keywords*—Minkowski sum; contributing vertices; 3D envelope computation**

1. Introduction

The Minkowski sum of two polyhedra in $\mathbb{R}^3$ is a fundamental task for many applications, such as computer-aided design and manufacturing [1], computer animation and morphing [2], morphological image analysis [3], robot motion planning [4], and solid modeling.

For two polyhedra A and B in $\mathbb{R}^3$, the Minkowski sum polyhedron S is the result of the position vector addition of all elements a and b coming from A and B respectively: $S = A \oplus B = \{a + b | a \in A, b \in B\}$.

A second definition states that the Minkowski sum of two polyhedra A and B is obtained by sweeping all points of A by B, i.e. translating B such that its origin (the common initial point of all its position vectors) passes through all boundary points of A, and taking the union of all resulting points (see Fig. 1):

$$A \oplus B = \bigcup_{a \in A} B_a \qquad (1)$$

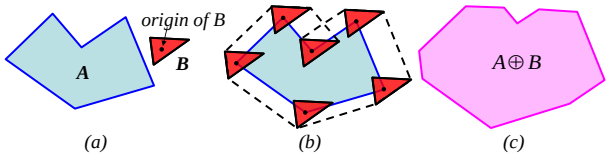

Fig. 1. (a) Two polygons A and B. (b) Sweeping all boundary points of A by B. (c) The Minkowski sum polygon $A \oplus B$.

Where $\cup$ denotes the set union operation and B_a denotes the set B translated by a position vector a.

A non-convex polyhedron is a polyhedron having one or more reflex edges. An edge is called a reflex edge if it exhibits a reflex angle, i.e. whose adjacent facets comprise an angle of more than 180 degrees w.r.t. (with respect to) the interior of the polyhedron [5]. We call a non-convex polyhedron **without fold** if its boundary is completely recoverable from three orthographic projections defined w.r.t. three orthogonal basis vectors in $\mathbb{R}^3$. Fig. 2.a illustrates in 2D the concept of a polygon without fold. Note that the polygon in Fig. 2.b has a fold because there are no two orthographic projections defined by two orthogonal basis vectors in $\mathbb{R}^2$ that allow the complete recovery of its boundary.

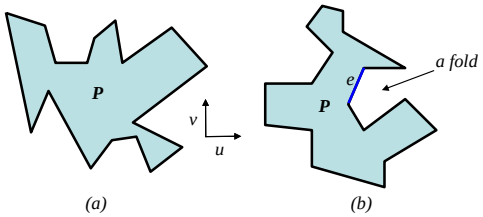

Fig. 2. (a) P is a non-convex polygon without fold. (b) P has a fold, the edge e can not be recovered completely from any two orthographic projections defined by an orthogonal basis in $\mathbb{R}^2$.

The Minkowski sum of a non-convex–convex pair of polyhedra can be seen as a general offset problem of a non-convex polyhedron and a convex one (which is not necessarily a sphere). In this work, we took benefit from the concept of contributing vertices we introduced in [6] and propose a novel algorithm for the computation of the Minkowski sum of a non-convex polyhedron without fold and a convex polyhedron. We show that we can generate a superset of the Minkowski sum facets and extract the exact Minkowski sum boundary by 3D envelope computation. Therefore, we avoid convex decomposition, union, and even convex hull computation.

The rest of this paper is organized as follows. A literature review is presented in section II. In section III, we describe briefly the Contributing Vertices-based Minkowski Sum (CVMS) algorithm for convex polyhedra. After that, we present in section IV an algorithm exploiting the concept of contributing vertices in order to generate a superset of the facets of the Minkowski sum of a non-convex–convex pair of polyhedra. In section V, we show how we can extract the Minkowski sum polyhedron from the superset of facets generated in the previous step. Finally, we present a complexity study, a performance benchmark, and some results.

2. Literature review

The Minkowski sum of two convex polyhedra A and B is obtained by performing the vector addition of all points of A and B and computing the convex hull of the resulting points. This process takes $O(mn)$ time for polyhedra with m and n features. For two non-convex polyhedra, the most common approach is based on convex decomposition of polyhedra. It takes $O(m^3n^3)$ time [7] to compute the sum by decomposing each non-convex polyhedron into convex pieces [5], computing the pairwise Minkowski sums of all possible pairs of convex pieces, and performing the union of the pairwise Minkowski sums [8]. The main bottleneck of this approach is the union step which is very time-consuming.

Another decomposition method was proposed by Evans et al. [9]. The authors decomposed the boundary of the summands into affine cells (instead of convex pieces), computed the pairwise Minkowski sums of transversal affine cells, and performed their union. The decomposition into affine cells yields more pieces than does the decomposition into convex pieces.

Recently, Hachenberger presented the first exact and robust implementation of a convex decomposition-based Minkowski sum algorithm [10] that performs in $O(m^3n^3)$ time for polyhedra with m and n features. He implemented an optimal convex decomposition algorithm similar to that proposed by Chazelle [5] and computed the union of the pairwise Minkowski sums through the use of Nef polyhedra [11].

To avoid drawbacks related to convex decomposition-based approaches, Varadhan and Manocha [12] approximated the union of the pairwise Minkowski sums using an adaptively subdivided voxel grid. They used isosurface extraction and guaranteed the same topology as the exact Minkowski sum. Recently, Lien [13] used a point-based representation instead of the mesh-based one. He constructed a point set by adding all points from two point sets uniformly sampled from the boundary of two polyhedra. He used three filters (a collision detection, normal, and octree filter) to discard inner points and showed several applications of the point-based representation. In the worst case (when the summands are convex), the point-based approach has a time complexity of $O(mnT_{filter})$, where m and n are the sizes of the point sets sampled from the summands and T_{filter} is the time complexity of filtering a single point.

Some dual space-based approaches have been investigated for the computation of the Minkowski sum. Ghosh [14] presented a unified computational framework for the Minkowski sum of polygons and polyhedra. Polyhedra are represented in a dual space—the slope diagram. The sum polyhedron results from the merging of two slope diagrams. The existent implementations of slope diagram-based algorithms work only for convex polyhedra. Other slope diagram variants can be found in [6], [15], [16], [17].

Guibas et al. [18], [19] defined the operation of convolution on planar tracings in 2D. Basch et al. [20] extended this definition to polyhedral tracings and used it to generate a superset of the Minkowski sum. They used arrangement computations to extract the exact boundary of the Minkowski sum. Their algorithm computes the convolution Q of two polyhedral tracings of size m and n in an output-sensitive time $O(k\alpha(k)log^3k)$, where k is the size of the convolution Q and $\alpha(k)$ is the familiar inverse Ackermann function. Nevertheless, at the best of our knowledge, the convolution-based approach for polyhedra has not been yet implemented.

3. Overview of the CVMS algorithm for convex polyhedra

Let A and B be two convex, closed, and two-manifold polyhedra. A is composed of f_A facets, e_A edges, and v_A vertices. B is composed of f_B facets, e_B edges, and v_B vertices. We stated in [6] that the Minkowski sum polyhedron $S = A \oplus B$ is a convex polyhedron composed of three types of facets: f_A facets with supporting planes parallel to those of the facets of A, these facets are named the "translated facets" of S; f_B facets with supporting planes parallel to those of the facets of B, these facets are named the "corner facets" of S; and at most e_Ae_B facets that result from the Minkowski sum of two non-parallel edges of A and B, these facets are named the "edge facets" of S.

The three categories of facets are obtained from the facets of A and B by computing their contributing vertices. The concept of contributing vertices is defined below and illustrated in Fig. 3. Further details can be found in [6].

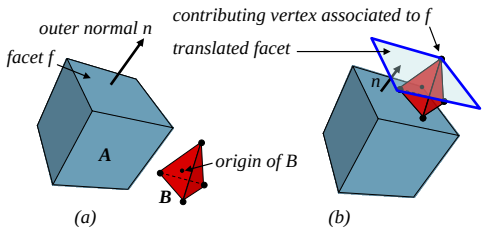

Fig. 3. The concept of contributing vertices. (a) Two convex polyhedra A and B. (b) The contributing vertex associated to a particular facet f of A is the vertex of B which is farthest from the supporting plane of f.

Definition 1: the **contributing vertex** $v_{k,B}$ of a particular facet $f_{i,A}$ with an outer normal $n_{i,A}$ is the vertex of B which is at maximal distance away from the supporting plane of $f_{i,A}$. Formally, the contributing vertex $v_{k,B}$ of a particular facet $f_{i,A}$ satisfies:

$$\langle v_{k,B}, n_{i,A} \rangle = \max_{\mathbb{R}} \langle v_{l,B}, n_{i,A} \rangle \ \forall l = 1, 2, ..., v_B \quad (2)$$

Where $\langle ., . \rangle$ denotes the scalar product.

The contributing vertex $v_{k,A}$ of a particular facet $f_{i,B}$ having an outer normal $n_{i,B}$ is defined in a similar manner by interchanging A and B in equation 2. For some facets of A and B, we can find many contributing vertices due to the fact that these contributing vertices are at the same (maximal) distance away from the supporting planes of the considered facets.

For each facet $f_{i,A}$, if it has one contributing vertex, the corresponding translated facet is obtained by a translation of $f_{i,A}$ according to a position vector starting from coordinates origin O and ending at the contributing vertex $v_{k,B}$ of $f_{i,A}$. If it has two contributing vertices, the corresponding translated facet is obtained by a planar Minkowski sum of $f_{i,A}$ and the edge of B incident to the two contributing vertices. Finally, if $f_{i,A}$ has more than two contributing vertices, the corresponding translated facet is obtained by a planar Minkowski sum of $f_{i,A}$ and the facet of B incident to the contributing vertices associated to $f_{i,A}$. The corner facets are obtained in the same manner from the contributing vertices associated to the facets of B. So the translated facets of $A \oplus B$ are corner facets of $B \oplus A$ and vice-versa.

The edge facets of the sum polyhedron are obtained by sweeping all edges of A by B. Because not all facets resulting from this sweep lie on the boundary, we used two criteria to retain only valid ones. So for each edge $e_{i,A}$, the first criterion is a visibility criterion. It enables to find horizon edges of B w.r.t. the direction defined by the edge $e_{i,A}$. The second criterion of normal orientation selects among the horizon edges, those that are valid for the computation of edge facets. We will revisit these two criteria in more depth in section IV-B.

We also gave two important properties. The first property states that if an edge of A or B is incident to facets having at least one common contributing vertex, then this edge will never contribute in any edge facets construction and can be simply ignored. The second property states that each facet from A or B contributes only once in the sum polyhedron. These two properties improve significantly the performance of the CVMS algorithm by reducing the number of facets and edges to be considered when computing translated, corner, and edge facets.

We have considered in the previous paragraphs that the origin of B coincides with the coordinates origin O. If the origin of B is any other point c in $\mathbb{R}^3$, then it suffices to take the Minkowski sum polyhedron computed with O as an origin of B and to translate it by a position vector starting from c and ending at O.

For convex polyhedra A and B, the CVMS algorithm has a worst-case time complexity of $O(f_A v_B + f_B v_A + f_A + f_B + e_A e_B)$.

4. Adaptation of the contributing vertices concept and generation of the Minkowski sum facets superset

In this section, we take benefit from the concept of contributing vertices and show how we can use it to develop a novel algorithm for the computation of a superset of the Minkowski sum facets of a non-convex–convex pair of polyhedra. Throughout the rest of the paper, A denotes the non-convex polyhedron and B denotes the convex polyhedron.

4.1 Adaptation of the CVMS properties for a non-convex–convex pair of polyhedra

The first property of the CVMS algorithm for convex polyhedra states that if an edge of A or B is incident to facets having at least one common contributing vertex, then this edge will never contribute in any edge facets construction and can be simply ignored. For non-reflex edges of A, this property remains true because the corresponding facets of the sum are translations of the two facets w.r.t. the common contributing vertex and are incident to the same edge of the sum polyhedron.

An important property of the facets which are incident to the same reflex edge is that their outer normals are oriented towards each other. Therefore, the corresponding translated facets which are generated by translations in the directions of their two outer normals collide. We conclude that there are no edge facets generated by reflex edges since they will necessarily lie inside the sum polyhedron.

Property 1: If two adjacent facets $f_{i,A}$ and $f_{j,A}$ are **either incident to a reflex edge** or they have at least one common contributing vertex, the edge $e_{k,A}$ shared by these adjacent facets will never contribute in any edge facets construction and can be simply ignored.

Property 1 is not applicable to any two adjacent facets $f_{m,B}$ and $f_{n,B}$ having at least a common contributing vertex (as done for convex polyhedra). Since A is a non-convex polyhedron, computing the contributing vertices associated to the facets of B is not convenient. A typical case occurs when two vertices $v_{i,A}$ and $v_{j,A}$ of A are contributing vertices of a particular facet $f_{k,B}$ but they are not incident to the same edge of A (this configuration is unfeasible for convex polyhedra but it is a valid one for non-convex ones). In this case, if we consider that the corner facet is the planar Minkowski sum of $f_{k,B}$ and the edge incident to $v_{i,A}$ and $v_{j,A}$ (as done in corner facets determination step for convex polyhedra), we will introduce a facet that is not by definition in the Minkowski sum polyhedron (since there is no edge of A connecting $v_{i,A}$ and $v_{j,A}$). So, we will not compute contributing vertices for the facets of B.

The second property of the CVMS algorithm for convex polyhedra states that each facet of A or B contributes only once in the Minkowski sum polyhedron S (a proof can be found in earlier works [2], [14], [15]). Suppose that the non-convex polyhedron A is decomposed into convex polyhedra $A_1, ..., A_n$. This implies that $A \oplus B = \bigcup_{i=1,...,n} (A_i \oplus B)$. Since there is no facet of A which appears twice in A_i and A_j (except facets introduced by the walls used when decomposing polyhedra), we conclude that each facet of A contributes only once in the Minkowski sum $A \oplus B$. In contrast, B (and thus each of its facets) appears in each pairwise Minkowski sums $A_i \oplus B$. So, each facet of B contributes more than once.

Property 2: Each facet of the non-convex polyhedron A contributes only once while each facet of the convex polyhedron B contributes more than once in $A \oplus B$.

After the two properties related to the contributing vertices concept have been adapted to our scenario, it remains to generate the translated, edge, and corner facets of $A \oplus B$.

4.2 Construction of the Minkowski sum facets superset

Since B is a convex polyhedron, the translated facets determination step is the same as for two convex polyhedra except that the facets of B contribute many times.

(1) **Translated facets determination:** for each facet $f_{i,A}$, compute its contributing vertices $v_{k,B}$ (see equation 2).

- If there are more than two contributing vertices, there exists a facet $f_{j,B}$ incident to the contributing vertices associated to $f_{i,A}$ and which lies on a supporting plane parallel to that of $f_{i,A}$. The corresponding translated facet is the planar Minkowski sum $f_{i,A} \oplus f_{j,B}$. The facet $f_{j,B}$ will be considered in the corner facets determination step since it contributes more than once (see property 2 above).
- If there are exactly two contributing vertices, there is an edge $e_{j,B}$ incident to the two contributing vertices associated to $f_{i,A}$ and which lies on a supporting line parallel to the supporting plane of $f_{i,A}$. The resulting translated facet is the planar Minkowski sum $f_{i,A} \oplus e_{j,B}$.
- If there is only one contributing vertex $v_{k,B}$, the corresponding translated facet is computed by translating the facet $f_{i,A}$ by a position vector starting at coordinates origin O and ending at $v_{k,B}$.

An example of a non-convex polyhedron A and a convex polyhedron B is depicted in Fig. 4.a and 4.b respectively. The translated facets of $A \oplus B$ are shown in Fig. 4.d.

For the edge facets determination step, we ignore reflex edges and consider the fact that no contributing vertices are computed for the facets of B (since A is a non-convex polyhedron).

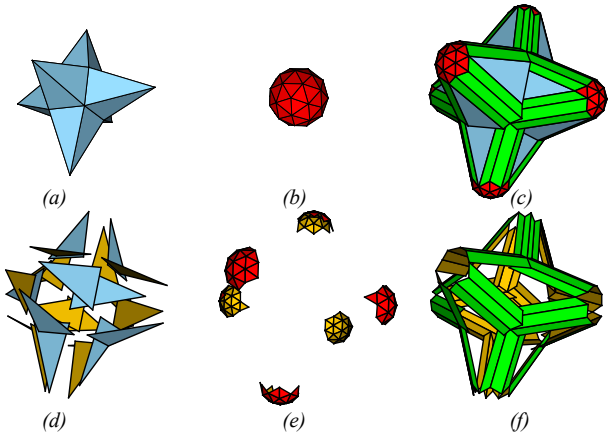

Fig. 4. (a) A non-convex polyhedron A. (b) A convex polyhedron B. (c) The entire superset of facets of the Minkowski sum $A \oplus B$. (d) The translated facets of $A \oplus B$. (e) The corner facets of $A \oplus B$. (f) The edge facets of $A \oplus B$.

(2) **Edge facets determination:** for each **non-reflex edge** $e_{i,A}$ incident to facets having distinct contributing vertices, find the horizon edges of B (edges separating invisible facets from other facets of B) w.r.t. the visibility direction $e_{i,A}$. This resumes to find invisible facets $f_{j,B}$ having outer normals $n_{j,B}$ that satisfy:

$$\langle e_{i,A}, n_{j,B} \rangle > 0 \tag{3}$$

- For each horizon edge $e_{k,B}$: an edge facet $e_{i,A} \oplus e_{k,B}$ is added to the sum polyhedron S if its outer normal orientation $n_{i,k} = e_{i,A} \times e_{k,B}$ lies between the two outer normal orientations $n_{1,A}$ and $n_{2,A}$ of the facets $f_{1,A}$ and $f_{2,A}$ incident to the edge $e_{i,A}$ ($n_{2,A}$ follows $n_{1,A}$ in a counterclockwise order). This is equivalent to: $\langle n_{1,A} \times n_{i,k}, e_{i,A} \rangle < 0$ and $\langle n_{i,k} \times n_{2,A}, e_{i,A} \rangle < 0$.

The edge facets of the example we considered above are depicted in Fig. 4.f.

Since A is a non-convex polyhedron, it is not possible to compute the contributing vertices associated to the facets of B in a global fashion. Therefore, we will consider that A is the union of several convex features and use the contributing vertices concept indirectly and locally throughout each convex feature. By convex features, we mean edges and vertices of A satisfying some conditions.

Definition 2: a non-reflex edge $e_{i,A}$ is called an **elevated edge** w.r.t. a direction defined by a vector u if u lies between the two outer normal orientations $n_{j,A}$ and $n_{k,A}$ of the facets $f_{j,A}$ and $f_{k,A}$ incident to the edge $e_{i,A}$ ($n_{j,A}$ follows $n_{k,A}$ in a counterclockwise order w.r.t a view along the oriented edge $e_{i,A}$ towards its end; see Fig. 5.a). Formally, an elevated edge $e_{i,A}$ w.r.t. a vector u satisfies:

$$\langle n_{j,A} \times u, e_{i,A} \rangle < 0 \;\; and \;\; \langle u \times n_{k,A}, e_{i,A} \rangle < 0 \quad (4)$$

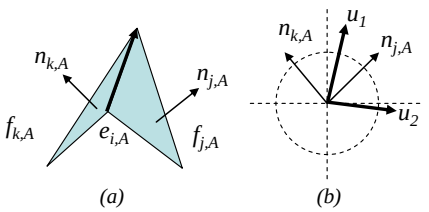

Fig. 5. (a) A non-reflex edge $e_{i,A}$. (b) A 2D view in the plane perpendicular to $e_{i,A}$. The edge $e_{i,A}$ is elevated w.r.t. u_1 but it is not an elevated edge w.r.t. u_2.

Definition 3: A vertex $v_{i,A}$ is called a **non-reflex vertex** if each of its incident edges are non-reflex edges.

Definition 4: A vertex $v_{i,A}$ is called an **elevated vertex** w.r.t. a direction defined by a vector u if it is a non-reflex vertex and if it satisfies the following equation:

$$\langle v_{i,A}, u \rangle = \max_{\mathbb{R}} \langle v_{j,A}, u \rangle \;\; \forall j = 1, 2, ..., |v_{i,A}| \quad (5)$$

Where $|v_{i,A}|$ denotes the valence of the vertex $v_{i,A}$, i.e. the number of edges incident to it. $v_{j,A}, j = 1, 2, ..., |v_{i,A}|$ are the 1-ring neighbors of $v_{i,A}$ (vertices sharing edges that are incident to $v_{i,A}$).

An elevated vertex is depicted in Fig. 6.a. If we take the opposite planes to the supporting planes of the facets $f_{k,A}, k = 1, 2, ..., |v_{i,A}|$ incident to the vertex $v_{i,A}$, then their intersection will define a frustum with apex $v_{i,A}$ (see Fig. 6.b). An equivalent definition of the elevated vertex can be formulated as follows: a vertex $v_{i,A}$ is an elevated vertex w.r.t. the direction defined by a vector u if u belongs to the frustum resulting from the intersection of the planes opposite to those supporting the facets incident to $v_{i,A}$.

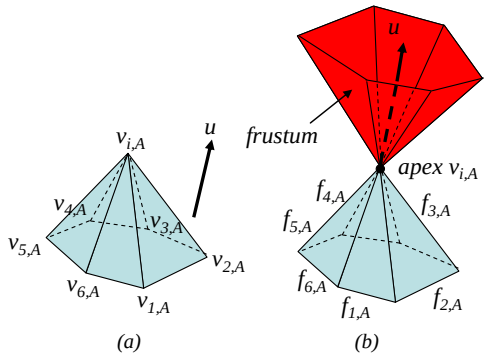

Fig. 6. (a) An elevated vertex $v_{i,A}$ w.r.t. the direction defined by the vector u. (b) A frustum created by the intersection of the planes opposite to those supporting the facets $f_{1,A}, f_{2,A}, ..., f_{6,A}$ incident to the vertex $v_{i,A}$. The vector u belongs to the created frustum.

Now, if we sweep B by A in order to compute corner facets, the features of A that are considered are those having normal orientations coinciding with the orientations of the outer normals of the facets of B. So, the corner facets determination step is as follows:

(3) **Corner facets determination:** for each facet $f_{i,B}$ of B having an outer normal $n_{i,B}$:

- If there is an edge $e_{j,A}$ of A which is elevated w.r.t. $n_{i,B}$ and which lies on a supporting line parallel to the plane supporting $f_{i,B}$, then the two vertices of A incident to $e_{j,A}$ are considered as local contributing vertices of $f_{i,B}$. The resulting corner facet is the planar Minkowski sum $f_{i,B} \oplus e_{j,A}$.
- If there is a vertex $v_{k,A}$ of A which is an elevated vertex w.r.t. $n_{i,B}$, the vertex $v_{k,A}$ is a local contributing vertex associated to $f_{i,B}$. So the corresponding corner facet is computed by translating the facet $f_{i,B}$ by a position vector pointing from coordinates origin O and ending at $v_{k,A}$.

Finally, the corner facets of our example are shown in Fig. 4.e. The entire superset of the Minkowski sum facets is depicted in Fig. 4.c.

5. Extraction of the Minkowski sum polyhedron from the superset of Minkowski sum facets

The process of construction of the superset of the Minkowski sum facets is valid for any non-convex–convex pair of polyhedra. A view from outside the superset of facets in any direction will always show a closed polyhedron. Nevertheless, the intersections among these facets must be handled in order to keep only portions of them lying completely on the boundary of the sum polyhedron.

In this work, we are restricted to the special class of non-convex polyhedra without fold we defined in section I. So, the process we will explain below is only valid for a non-convex polyhedron without fold A and a convex polyhedron B.

This definition of polyhedra without fold characterizes a class of non-convex polyhedra that can be projected into three planes having u, v, and w as outer normals such that each facet can be completely recovered from these orthographic projections. Note that the star-shaped polyhedra class is a subset of the non-convex polyhedra without fold class (the non-convex polygon without fold P in Fig. 2.a is not a star-shaped polygon because there is no point $a \in P$ such that the line segment $ab \subset P \mid \forall b \in P$).

To extract the Minkowski sum facets from the superset already generated, it is sufficient to compute the lower and upper 3D envelopes of the superset of facets w.r.t. each basis vector u, v, and w, and to take the union of the resulting envelopes. This guarantees the complete recovery of all facets of the Minkowski sum polyhedron S and at the same time, the handling of the intersections and the elimination of patches of facets that do not lie on the boundary of S (they will not be visible from outside S).

The computation of lower and upper envelopes involves the computation of minimization and maximization diagrams. The minimization (or maximization) diagram for a set σ of xy-monotone surfaces is the subdivision of the xy-plane into cells, such that the identity of the surfaces that induce the lower (or upper) diagram over a specific cell of the subdivision is the same. Further details about lower and upper envelopes can be found in [21].

The principle of our Minkowski sum boundary extraction approach is revealed by algorithm 1. It is explained in Fig. 7 for 2D polygons. In 2D, we speak about x-monotone curves (intersecting each line parallel to the y-axis at a single point at most) and their envelopes. So, the minimization and maximization diagrams are subdivisions of the x-axis. Fig. 7.a depicts a typical superset of edges (with self intersections) of the Minkowski sum of a non-convex polygon without fold and a convex polygon. In order to extract the boundary of the Minkowski sum polygon, we consider the two orthogonal vectors u and v. In Fig. 7.b, the superset of faces (or edges of the polygon) is rotated by $90°$ counterclockwise in order to have u coincident with y-axis and the lower and upper envelopes are computed. At this stage, some parts of the boundary of the Minkowski sum are not yet extracted from the superset of edges. The computed envelopes are rotated again to return to their original positions (Fig. 7.c). In Fig. 7.d, no rotation of the superset of edges is needed since v coincides with y-axis, the lower and upper envelopes are also computed. Finally, the last step is to perform the merge of the lower and upper envelopes computed w.r.t. u and v (see Fig. 7.e) and to compute their union in order to obtain the Minkowski sum polygon depicted in Fig. 7.f.

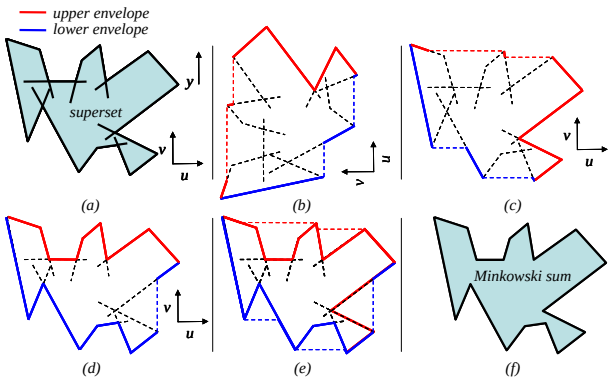

Fig. 7. An illustration of the Minkowski sum boundary extraction algorithm in 2D.

Because the envelopes computation time depends on the number of surfaces involved, we use a trick to split the superset of facets into two sets, one of them is used for the lower envelope computation and the other for the upper envelope computation. These two sets are denoted *lower candidates* and *upper candidates* respectively. Since the Minkowski sum polyhedron we want to extract is closed and since the lower and upper envelopes are always contiguous planar maps (without holes), a facet f_i having outer normal orientation n_i oriented outside the considered z-axis (u, v, or w) will be hidden by other facets having outer normals oriented towards the z-axis when computing the upper envelope. So, ignoring this facet will eliminate some computation overhead without affecting correctness of the overall algorithm.

Therefore, if we want to compute the envelopes when u is coincident with the z-axis, we put a particular facet f_i in the upper candidates set if its outer normal n_i is oriented towards u, i.e. $\langle n_i, u\rangle > 0$. Similarly, we put f_i in the lower candidates set if its outer normal n_i is oriented outwards u, i.e. $\langle n_i, u\rangle < 0$. If the outer normal n_i of a particular facet f_i is perpendicular to the considered z-axis ($\langle n_i, z\rangle = 0$), we simply ignore this facet from both lower and upper envelopes computation since it will be automatically recovered when considering the other z-axes. The same reasoning is applied when v or w are considered as z-axis.

So, given a superset $F = (f_1, f_2, \ldots, f_n)$ of the Minkowski sum facets of a non-convex polyhedron A without fold and a convex polyhedron B, we denote the upper and lower candidates as C_{upper} and C_{lower} respectively. $U(C_{upper})$ and $L(C_{lower})$ denote the computed upper envelope for C_{upper} and the computed lower envelope for C_{lower} respectively. $V = \{u, v, w\}$ denotes the set of three orthogonal vectors used for computing envelopes. The algorithm of the extraction of Minkowski sum polyhedron S from F is as follows:

Algorithm 1 Extraction of the Minkowski sum boundary from the superset of facets

Require: a superset of the Minkowski sum facets F and a set of basis vectors V
Ensure: the Minkowski sum polyhedron S
for each basis vector $v_i \in V, i = 1, 2, 3$ **do**
  clear lower and upper candidates C_{upper} and C_{lower}
  get C_{upper} and C_{lower}
  compute the upper envelope $U(C_{upper})$
  compute the lower envelope $L(C_{lower})$
  if S is empty ($i = 1$) **then**
    add the upper envelope $U(C_{upper})$ to S
    add the lower envelope $L(C_{lower})$ to S
  else
    compute the union of S and the upper envelope $U(C_{upper})$ ($S \leftarrow S \cup U(C_{upper})$)
    compute the union of S and the lower envelope $L(C_{lower})$ ($S \leftarrow S \cup L(C_{lower})$)
  end if
end for

6. Implementation and performance

In this section, we describe our algorithm's implementation. We also give a complexity study, a performance benchmark, and present some results.

TABLE 1
PERFORMANCE BENCHMARK OF OUR ALGORITHM AND THE NUMBER OF FACETS OF THE MINKOWSKI SUM COMPUTED FOR SEVERAL POLYHEDRA.

Operands (facets number)		Runtime(sec.)			
A (non-convex without fold)	B (convex)	Superset computation	Envelope extraction	Overall time	Number of facets of $A \oplus B$
Star(24)	IcosiDodeca.(32)	0.172	6.953	7.125	320
L(20)	Sphere3(320)	0.391	10.250	10.641	527
Star(24)	Sphere3(320)	0.454	29.094	29.548	1120
Rabbit(2000)	Sphere3(320)	25.188	384.937	410.125	6155
Rabbit(10000)	Sphere2(80)	34.782	1163.924	1198.706	11225

6.1 Implementation

For the implementation of our algorithm, we used C++ and CGAL [22]. The exactness of the result is guaranteed through the use of the exact number types provided by the GNU Multi Precision (GMP) library [23]. We also used the lazy kernel adapter [24], which speeds up exact computations.

6.2 Complexity study

The first part of our algorithm aims at producing a superset of the Minkowski sum facets. So, the computation of the contributing vertices for the f_A facets of the non-convex polyhedron A requires $O(f_A v_B)$ time. The determination of translated facets is done directly from the computed contributing vertices in $O(f_A)$ time. The computation of the edge facets takes at most $f(e_A e_B)$ time since not all edges will be considered. Finally, the computation of the corner facets requires $O(f_B v_A + f_B e_A)$ time. Note that the enumeration of the elevated vertices of A w.r.t. a direction defined by the outer normal of a particular facet of B is linear to the number of vertices of A and the maximum valence on the vertices of A (this factor is considered constant and does not appear in the time complexity of the corner facets determination step). Taking the sum of these partial time complexities leads to a worst-case time complexity of $O(f_A v_B + f_A + f_B v_A + f_B e_A + e_A e_B)$ for the computation of the superset of Minkowski sum facets.

The complexity of the lower/upper envelope of a set of surfaces is defined as the complexity of its minimization/maximization diagram [21]. For n surface patches in $\mathbb{R}^3$, it takes at most $O(n^{2+\epsilon})$ time to compute their lower or upper envelope (for any $\epsilon > 0$) [21]. Therefore, the complexity of the CVMS algorithm is the addition of the complexities of the superset facets construction and the envelope computation. So, it takes $O(f_A v_B + f_A + f_B v_A + e_A e_B + f_F^{2+\epsilon})$ time to compute $A \oplus B$ where f_F is the number of facets of the superset. Note that the combinatorial complexities of the union operation, the envelope rotation, and the lower/upper candidates computation are ignored in comparison to the combinatorial complexity of the envelope computation.

Although the quadratic complexity of the CVMS algorithm may seem high, it should be viewed in context of the high complexity of the convex decomposition-based approach ($O(v_A^3 v_B^3)$) which has been recently implemented [10].

6.3 Performance benchmark and results

The experiments where done on a 2 GB RAM, 2.2 GHZ Intel Core 2 Duo personal computer. Table 1 gives some results and run-times of our algorithm on several polyhedra. Some models computed by our algorithm are depicted in Fig. 8.

Table 1 indicates that our algorithm works well with small as with larger models having few thousand facets or tens of thousands of facets. The run-times are reasonable when considering the use of exact number types which are slower than built-in ones. As an example, our algorithm computes the Minkowski sum of a rabbit model having 2000 facets and a sphere having 320 facets in less than 7 minutes (410.125 seconds) and the Minkowski sum polyhedron has 6155 facets. The same computation took 22 minutes (1293.94 seconds) when performed with the convex decomposition implementation of Hachenberger [10]. The computation of the union of the 3D envelopes is the most time consuming step in the process. In contrast, the generation of the superset of Minkowski sum facets is a quick process because the only facets computed in this step are those which are relevant for the Minkowski sum boundary extraction.

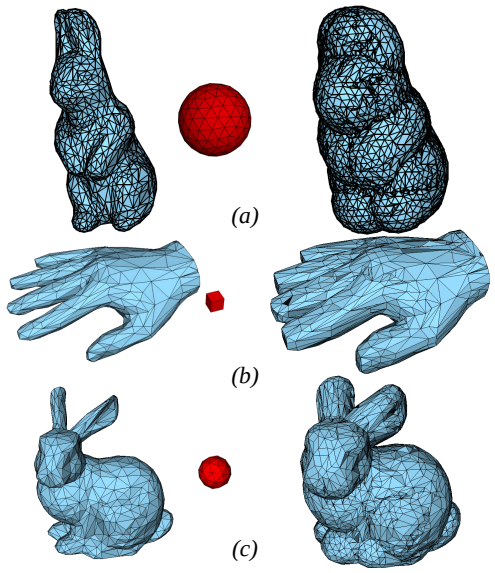

Fig. 8. Minkowski sum examples. From left to right: the polyhedron A, the polyhedron B, and the Minkowski sum polyhedron $A \oplus B$. (a) The Minkowski sum (6155 facets) of the rabbit model (2000 facets) and a sphere (320 facets). (b) The Minkowski sum (1893 facets) of the hand model (1000 facets) and a cube (6 facets). (c) The Minkowski sum (4440 facets) of the bunny model (1500 facets) and a sphere (80 facets).

7. Conclusion and future work

We have presented a novel algorithm based on the concept of contributing vertices for the computation of the Minkowski sum of a non-convex-polyhedron without fold A and a convex polyhedron B. First, we computed a superset of the Minkowski sum facets of A and B. Secondly, we extracted the boundary of the Minkowski sum polyhedron from the superset of facets by performing the union of 3D envelopes computed from three orthogonal axes. The implementation of our algorithm guarantees the exactness of the results. It computes the Minkowski sum of a non-convex polyhedron without fold and a convex polyhedron without decomposition and union steps.

One of the drawbacks of our algorithm is that there is a redundancy in the computation of envelopes. It will be better if we find a criterion eliminating facets completely recovered at each step from further computations in order to reduce computation time. Actually, we have no way to determine the u, v, and, w directions. However, we done several experiments which confirmed that nearly all u, v, and, w directions are suitable for the extraction of the Minkowski sum boundary. For some specific models, the sum of a non-convex polyhedron without fold and a convex polyhedron gives a polyhedron having folds, which can not been handled by our algorithm.

As a part of our future work, we are aiming to generalize our algorithm to non-convex polyhedra having folds. We are investigating two ideas: the first is the computation of several local envelopes for disjoint subsets of facets and the assembly of the local envelopes. The second idea is to use some region growing techniques to start from seed facets and explore their neighborhood until the Minkowski sum polyhedron is obtained.

Acknowledgments

This work is partly supported by the Cluster ISLE of the Rhône-Alpes region within the LIMA Project. We thank the reviewers for their observations and remarks that helped us to improve the paper.

References

[1] I-K Lee, M-S Kim, G Elber. Polynomial/rational approximation of Minkowski sum boundary curves. Graph. Models Image Process. 1998, 60(2): 136-165.

[2] A Kaul, J Rossignac. Solid-interpolating deformations: construction and animation of PIPs. Comput. Graph. 1992, 16(1): 107-115.

[3] J Serra. Image Analysis and Mathematical Morphology. Vol. 2. New York: Academic Press, 1988.

[4] T Lozano-Pérez. Spatial planning: a configuration space approach. IEEE Trans. Comput. 1983, 32(2): 108-120.

[5] B Chazelle. Convex decompositions of polyhedra. In: STOC '81: Proc. of the Thirteenth Annual ACM Symposium on Theory of Computing, ACM, New York: NY, USA, 1981: 70–79.

[6] H Barki, F Denis, F Dupont. Contributing vertices-based Minkowski sum computation of convex polyhedra. In revision, Comput. Aided Des..

[7] D Halperin. Robust geometric computing in motion. Int. J. Robotics Res. 2002, 21(3): 219-232.

[8] B Aronov, M Sharir, B Tagansky. The union of convex polyhedra in three dimensions. SIAM J. Comput. 1997, 26(6): 1670-1688.

[9] R Evans, M O'Connor, J Rossignac. Construction of Minkowski sums and derivatives morphological combinations of arbitrary polyhedra in CAD/CAM systems. US Patent 5159512, 1992.

[10] P Hachenberger. Exact Minkowski sums of polyhedra and exact and efficient decomposition of polyhedra in convex pieces. In: Proc. 15th Annual European Symposium on Algorithms, 2007: 669-680.

[11] P Hachenberger, L Kettner, K Mehlhorn. Boolean operations on 3d selective Nef complexes: data structure, algorithms, optimized implementation and experiments. Comput. Geom. Theory Appl. 2007, 38(1-2): 64-99.

[12] G Varadhan, D Manocha. Accurate Minkowski sum approximation of polyhedral models. Graph. Models 2006, 68(4): 343-355.

[13] J-M Lien. Point-based Minkowski sum boundary. In: PG '07: Proc. of the 15th Pacific Conference on Computer Graphics and Applications (PG'07), IEEE Computer Society, Washington, DC, USA, 2007: 261-270.

[14] P Ghosh. A unified computational framework for Minkowski operations. Comput. Graph. 1993, 17(4): 357-378.

[15] H Bekker, J Roerdink. An efficient algorithm to calculate the Minkowski sum of convex 3d polyhedra. In: ICCS '01: Proc. of the International Conference on Computational Sciences-Part I, Springer-Verlag, London, UK, 2001: 619-628.

[16] Y Wu, , J Shah, J Davidson. Improvements to algorithms for computing the Minkowski sum of 3-polytopes. Comput. Aided Des. 2003, 35(13): 1181-1192.

[17] E Fogel, D Halperin. Exact and efficient construction of Minkowski sums of convex polyhedra with applications. Comput. Aided Des. 2007, 39(11): 929-940.

[18] L J Guibas, L Ramshaw, L Stolfi. A kinetic framework for computational geometry. In: Proc. of 24th annual IEEE Symposium on the Foundation of Computer Science, 1983: 100-111.

[19] L Guibas, R Seidel. Computing convolutions by reciprocal search. Discrete Comput. Geom. 1987, 2: 175-193.

[20] J Basch, L Guibas, G Ramkumar, L Ramshaw. Polyhedral tracings and their convolution. In: Proc. of 2nd Workshop on the Algorithmic Foundations of Robotics, 1996: 171-184.

[21] M Meyerovitch. Robust, generic and efficient construction of envelopes of surfaces in three-dimensional space. In: Proc. 14th Annual European Symposium on Algorithms (ESA), Vol. 4168 of LNCS, Springer-Verlag, 2006: 792-803.

[22] The CGAL project homepage. http://www.cgal.org/.

[23] The GNU MP bignum library. http://gmplib.org/.

[24] A Fabri, S Pion. A generic lazy evaluation scheme for exact geometric computations. In: Proc. 2nd Library-Centric Software Design, 2006.