Extracting the Minkowski Sum Boundary from the Reduced Convolution *

Evan Behar Jyh-Ming Lien †

Abstract

We propose a new method for computing the 2-d Minkowski sum of non-convex polygons. Our method is convolution based. The main idea is to use the *reduced convolution* and filter the boundary by using the topological properties of the Minkowski sum. The main benefit of this proposed approach is from the fact that, in most cases, the complexity of the complete convolution is much higher than the complexity of the final Minkowski sum boundary. Therefore, the traditional approach often wastes a large portion of the computation on computing the arrangement induced by the complete convolution that is later on thrown away. Our method is designed to specifically avoid this waste of computation. We demonstrate both theoretically and experimentally that the proposed method handles significantly less number of line segments using only reduced convolution than the traditional approaches that use the complete convolution.

1 INTRODUCTION

The Minkowski sum (M-sum) plays important roles in many subfields of robotics. For example, it is the basic operation for mapping the workspace to the configuration space, and is fundamental in many geometric applications such as estimating the penetration depth between two intersecting objects. The M-sum of two shapes P and Q is defined as:

$$P \oplus Q = \{p + q \mid p \in P, q \in Q\}. \quad (1)$$

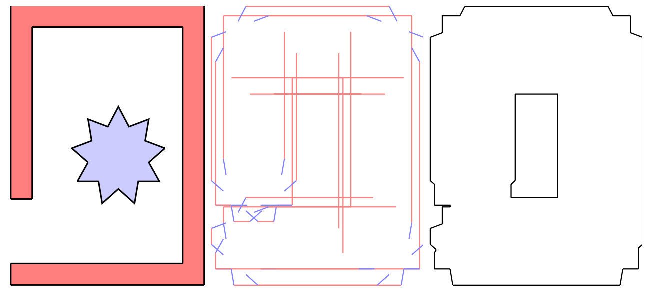
The idea of using M-sums to solve these problems has been proposed for more than three decades and many methods have been proposed to compute M-sums. However more efficient methods to compute both 2-d and 3-d M-sums have only been proposed recently. Surveys can be found in [Ghosh 1993; Varadhan and Manocha 2006; Fogel and Halperin 2006]. In particular for both 2-d and 3-d, several methods [Kaul and Rossignac 1991; Fogel and Halperin 2006; Gritzmann and Sturmfels 1993; Fukuda 2004] are known to compute the M-sum of *convex* polyhedra efficiently.

For computing the M-sum of non-convex shapes, many methods are based on the idea of convex decomposition. In these methods, the input models are decomposed into components. Because computing the M-sum of convex shapes is easier than non-convex shapes, convex decomposition is widely used. The next step in this framework computes the pairwise M-sums of the components. Finally, all these pairwise M-sums are united to form the final M-sum. Although conceptually simple, this method is usually not practical due to the size of the decomposition and the difficulty in implementing a robust union operation.

Convolution-based methods do not have these problems. The convolution of two shapes P and Q, denoted as $P \times Q$, is a set of line segments in 2-d or facets in 3-d that is generated by "combining" the segments or the facets of P and Q [Guibas et al. 1983]. One can think of the convolution as the M-sum that involves only the boundary, i.e., $P \times Q = \partial P \oplus \partial Q$. It is known that the convolution forms a superset of the M-sum [Ghosh 1993], i.e., $\partial(P \oplus Q) \subset P \times Q$. To obtain the M-sum boundary, it is necessary to trim the line segments or the facets of the convolution.

*This work is supported in part by NSF IIS-096053 and Autodesk.

†Both authors are with Department of Computer Science, George Mason University, 4400 University Drive MSN 4A5, Fairfax, VA 22030 USA, ebehar@gmu.edu, jmlien@cs.gmu.edu



(a) (b) (c)

Figure 1: *(a) Input polygons: star and fence. (b) The reduced convolution of star and fence. (c) The Minkowski sum of star and fence generated from the reduced convolution.*

1.1 Our Contribution

We propose a new method for computing the 2-d Minkowski sum (M-sum) of non-convex polygons. Our method can be viewed as a convolution-based approach. The main idea is to use the *reduced convolution* (defined later in Section 4) and extract the boundaries by using the topological properties of the M-sum. Fig. 1 illustrates the proposed approach.

The main idea of the reduced convolution is inspired by the fact that, in most cases, the complexity of the complete convolution is much higher than the complexity of the final M-sum boundary. Computing the arrangement induced by the complete convolution is expensive and many elements in this arrangement are later on thrown away.

To improve the efficiency, an obvious approach is to avoid computing the complete convolution and its arrangement. However, the difficulty now becomes whether we can define a smaller set of the convolution while still being able to extract the M-sum.

Our method is designed to specifically avoid this waste of computation and address these difficulties. A detailed description of the proposed method can be found in Section 4.

2 RELATED WORK

During the last three decades, many methods have been proposed to compute the Minkowski sums (M-sum) of polygons or polyhedra. Despite the large volume of work, most methods can be categorized into one of the two main frameworks: divide-and-conquer and convolution. Here we briefly review these techniques. A more detailed review on the M-sum can be found in our previous work [Lien 2008].

Divide-and-Conquer. This approach is first proposed by [Lozano-Pérez 1983] to compute $\mathcal{C}$-obst for motion planning. Although the main idea of this approach is simple, the divide step (i.e., convex decomposition) and the merge step (i.e., union) can be very difficult to implement robustly in practice, in particular when the input shapes are complex. For example, it is known that creating solid convex decomposition robustly is difficult, e.g., it is necessary to maintain the 2-manifold property after the split.

In addition, [Agarwal et al. 2000] have shown that different decomposition strategies can greatly affect the efficiency of this approach. Moreover, they show that the efficiency of any particular decomposition technique depends greatly on the input polygons.

The union step is even more troublesome. The decomposition step normally generates many components. No existing union method can robustly compute the union of thousands or even millions of pairwise M-sums.

In particular, the size and the complexity of the geometry generated during the intermediate steps can be overwhelming. [Flato 2000] computes the unions using the cells induced by the arrangement of the line segments. He uses a hybrid strategy that combines arrangement with incremental insertion to gain better efficiency.

Convolution. For 2-d polygons, [Guibas and Seidel 1987] show an output sensitive method to compute convolution curves. Later, [Ghosh 1993] proposed an approach, which unifies 2-d and 3-d, convex and non-convex, and M-sum and decomposition operations. The main idea in his method is the negative shape and slope diagram. The slope diagram is closely related to the *Gaussian map*, which has been recently used by [Fogel and Halperin 2006] to implement robust and efficient Minkowski sum computation of convex objects.

[Kaul and Rossignac 1991] proposed a linear time method to generate a set of M-sum facets. Output sensitive methods that compute the M-sum of polytopes in d-dimensions have also been proposed by [Gritzmann and Sturmfels 1993] and [Fukuda 2004].

The main difficulty of the convolution-based methods is to remove the portion of the facets that are inside the M-sum. [Barki et al. 2009] demonstrate an exact convolution-based method on restricted classes of summands in 2-d predicated on recovering the boundary from a set of orthographic projections in $\mathbb{R}^3$.

Recently, Wein [2006] showed a robust and exact method for non-convex polygons. To obtain the M-sum boundary from the convolution, his method computes the arrangement induced by the line segments of the convolution and keeps the cells with non-zero winding numbers.

3 PRELIMINARIES

In this section, we define the notations that will be used throughout the paper. Let P and Q be simple polygons composed of n and m (counterclockwise) ordered vertices, respectively. We denote the vertices of P as $\{p_i\}$ and the edge that starts at vertex p_i as $e_i = \overline{p_i p_{i+1}}$. The edge e_i has two associated vectors, the vector from p_i to p_{i+1}, i.e., $\vec{v_i} = \overrightarrow{p_i\, p_{i+1}}$, and the outward normal $\vec{n_i}$. The definition for the vertices $\{q_j\}$ and edges of Q is the same. Fig. 2 shows an example of P and Q.

The boundary of the Minkowski sum (M-sum) of P and Q is known to have complexity $O(m^2n^2)$ [Halperin 2002] and is composed of an external boundary and a (possibly empty) list of hole boundaries. Our approach for computing the M-sum boundary is based on the idea of convolution. The convolution of two shapes P and Q, denoted as $P \otimes Q$, is a set of line segments in 2-d that is generated by "combining" the segments of P and Q [Guibas et al. 1983]. One can think of the convolution as the M-sum that involves only the boundary, i.e., $P \otimes Q = \partial P \oplus \partial Q$. It is known that the convolution forms a superset of their M-sum boundary [Ghosh 1993], i.e., $\partial(P \oplus Q) \subset P \otimes Q$. If both P and Q are convex, $\partial(P \oplus Q) = P \otimes Q$. Otherwise, it is necessary to trim the convolution to obtain the M-sum boundary.

Specifically, an edge $\overline{p_i p_{i+1}}$ of P and a vertex q_j of Q (or vice versa) form a segment of $P \otimes Q$ if $\overrightarrow{p_i\, p_{i+1}} \in [\overrightarrow{q_{j-1}\, q_j}, \overrightarrow{q_j\, q_{j+1}})$, and we say that $\overline{p_i p_{i+1}}$ and q_j are *compatible*. Equivalently, $\overline{p_i p_{i+1}}$ and q_j are compatible if the outward normal of $\overline{p_i p_{i+1}}$ lies between the normals of the incident edges of q_j. For example, in Fig. 2, $\overline{p_3 p_1}$ and q_1 are compatible.

4 OUR METHOD

We propose a new method to compute the M-sum of simple non-convex polygons. Similar to Wein's method [Wein 2006], our method can be considered as a type of convolution-based approach. However, unlike Wein [2006], the proposed method avoids computing (1) the complete convolution, (2) the arrangement of the segments of the convolution, and (3) the winding number for each arrangement cell.

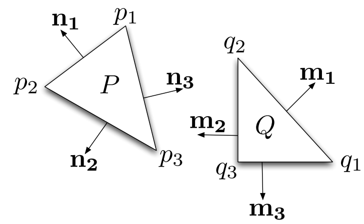


Figure 2: *Two convex polygons P and Q shown with the edges' outward normals.*

Algorithm 4.1: M-SUM(P, Q)

$R =$ Compute the reduced convolution of P and Q
$L =$ Extract orientable loops from R
$M =$ Filter boundaries from L
return (M)

Our method is sketched in Algorithm 4.1. Algorithm 4.1 first computes a subset of the segments that is from the convolution of the inputs. We call this subset a "reduced convolution."

Definition 4.1. *A* **reduced convolution** *is a set of segments $\overline{p_i p_{i+1}} \oplus q_j$ and $p_k \oplus \overline{q_l q_{l+1}}$ and q_j and p_k must be convex.*

Then Algorithm 4.1 identifies closed loops that are (1) non-overlapping and (2) *orientable*. These loops form potential boundaries of the M-sum and are further filtered by analyzing their nesting relationship. Finally, the remaining boundaries are filtered by checking the intersections between the input polygons placed at the configurations along these loops.

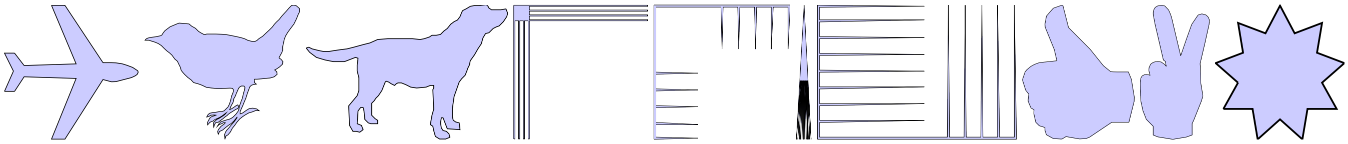
4.1 Reduced convolution

In the first step of the algorithm, we compute a subset of the segments of the convolution based on the following simple observation.

Observation 4.2. *Given a convolution segment $s = e_i \oplus q$ of an edge $e_i \in P$ and a vertex $q \in Q$, if q is a reflex vertex, s must not be a boundary of the M-sum of P and Q. This observation remains true if $s = p \oplus e_j$, where $p \in P$ is a reflex vertex and $e_j \in Q$ is an edge.*

Proof. Sketch. Let $\mathcal{S}$ be a set of segments formed by the end points of e_i and the edges incident to q. Because s must be incident to the segments $\mathcal{S}$, the vertex incident to both s and $\mathcal{S}$ is locally non-manifold. Moreover, by definition of convolution, s must be enclosed by the turning range of $\mathcal{S}$. Therefore, s cannot be on the boundary of the M-sum. □

Because of the definition of a reflex angle, the number of edges that are compatible with any convex vertex in Q form a lower bound on the number of edges compatible with any reflex vertex in Q. Due to this, the number of segments filtered by Observation 4.2 is significant.



(a) airplane (b) bird (c) dog (d) G1 (e) G2 (f) (g) G4 (h) hand1 (i) (j) star

Figure 3: *Models used in the experiments. The table shows the names and the sizes of the polygons. Note that (f) is G3, and (i) is hand2.*

4.2 Orientable Loop Extraction

Now, since the segments that we will be working with are no longer a complete convolution, we cannot apply the idea of computing the winding number for each arrangement cell to extract the M-sum boundary as done in [Wein 2006]. Instead, we proceed by defining two filters.

Observation 4.3. *We observe that the boundary of the Minkowksi sum must be an orientable loop (if it encloses an area, either positive or negative).*

We say that a loop is orientable if all the normal directions of the edges in the loop are all either pointing inward or outward. Note that the segments we considered are edges from P and Q, therefore, they are directional (as vertices in P and Q are ordered) and include normal directions pointing outward (to P or Q). Therefore, given two adjacent segments $s = \{u, v\}$ and $s' = \{v, u'\}$ sharing an end point v, we can check whether s and s' belong to an orientable loop if

$$\overrightarrow{uv} \times \vec{n_s} = \overrightarrow{vu'} \times \vec{n_{s'}} \ , \qquad (2)$$

where $\vec{n_x}$ is the normal vector of segment x, and $\times$ is the cross product. If s and s' satisfy Eq. 2, we say they are compatible segments.

To extract all orientable loops, we compute the intersections of the segments and split all segments at the intersections. A loop is then traced by starting at an arbitrary segment s that has not been considered and then iteratively including compatible segments adjacent to s. Note that there can be multiple compatible segments adjacent to s and all are incident to a single point v. This problem is in fact easy to handle since all M-sum boundaries must be manifold. Thus, we simply pick the segment that makes the largest clockwise turn from s among all the incident segments.

Observation 4.4. *The loops must obey the nesting property, i.e., the loops that are* directly *enclosed by the external loop must be holes and will have negative areas, and the loops that are directly enclosed by the holes must have positive areas.*

This is because all loops we generated are non-overlapping (i.e., they don't intersect or touch) due to the manifold properties. The nesting property can be determined efficiently using a plane sweep algorithm, e.g., [Bajaj and Dey 1990], in $O(n \log n)$ time for n segments.

4.3 Boundary Filtering

So far, we have introduced three quite efficient filters based on Observations 1 through 3. Unfortunately, not all of the remaining loops are boundaries of the M-sum. Therefore, we will have to resort to collision detection to remove all the false loops. Given a translational robot P and obstacles Q, the contact space of P and Q can be represented as $\partial((-P) \oplus Q)$, where $-P = \{-p \mid p \in P\}$. If a point x is on the boundary of the M-sum of two polygons P and Q, then the following condition must be true:

$$(-P^\circ + x) \cap Q^\circ = \emptyset \ ,$$

where Q° is the open set of Q and $(P + x)$ denotes translating P to x.

Although there are many methods to optimize the computation time for collision detection, collision detection is more time consuming than the previous filters. Fortunately, only a single collision detection is needed to reject or accept a loop based on the following lemma.

Lemma 4.5. *All the points on a false hole loop must make P collide with Q.*

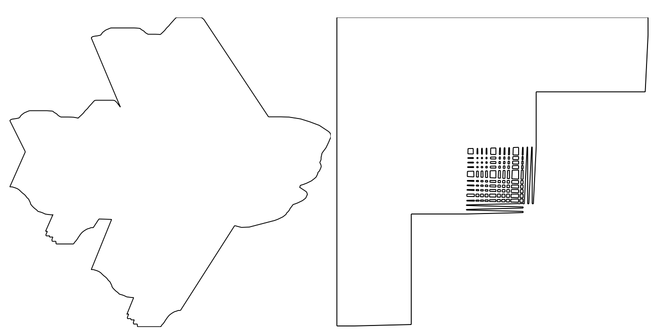
Proof. Sketch. Each loop must belong to a cell from the arrangement of the segments in the complete convolution. Moreover, all vertices in an arrangement cell must have the same winding number according to [Wein 2006]. Therefore, a single point from each loop is sufficient to test if the loop is a true boundary or not. □

4.4 Complexity Analysis

When P and Q have n and m vertices which include n' and m' reflex vertices, respectively, there will be $2mn$ segments in the complete convolution; in the reduced convolution there are at most $(m - m')n + (n - n')m$ segments. That is, the arrangement of the reduced convolution is at least 4 times less complex than that of the complete convolution when $n' = 1/2n$ and $m' = 1/2m$. Note that this analysis is based on the assumption that a convex vertex is compatible with $\Theta(n)$ edges and in the worst case that each segment will intersect all the other segments. In the examples that use in the experiment, the difference between the reduced and complete convolutions is more significant (e.g., "star/star" and "dog/bird"). The time complexity for computing the M-sum of P and Q is $O\left((mn + I)\log(mn + I) + \ell T_{cd}\right)$, where $I = O(m^2n^2)$ is the complexity of the arrangement of the reduced convolution, ℓ is the number of loops, and $T_{cd} = O(mn)$ is the collision detection time in our implementation.

5 EXPERIMENTAL RESULTS

The proposed method is implemented in C++ and we use the models in Fig. 3 as our test examples. In Fig. 4, we show two examples generated by the proposed method.



(a) bird⊕airplane (b) G1⊕G2

Figure 4: *Examples of the M-sums generated by our program using the models in Fig. 3. (a) There are 1339 vertices and an external boundary. (b) There are 1204 vertices and 101 boundaries.*

In addition, we show that the size of the reduced convolution is the key that the proposed method is more efficient. Theoretically, the reduced convolution is at most half of the complete convolution. However, this analysis (in Section 4.4) is based on the assumption that a convex vertex is compatible with $\Theta(n)$ edges. Practically, this assumption may be off. Fig. 5 shows exactly this. Again, we use the models in Fig. 3 and compute the M-sums of all pairs. We study the differences when the reduced convolution and the complete convolution are used.

Fig. 5 clearly shows that the size of the reduced convolution is significantly smaller than that of the regular convolution. Note that the y axis in this plot is in logarithmic scale. In the best case (bird$\oplus$dog), the reduced convolution is 13.13 times smaller. In this case, the reduced convolution has 2,921 segments and the convolution has 38,342 segments. In the worst case (G3$\oplus$G3), the reduced convolution is only 1.98 times smaller. In this case, reduced convolution has 320 segments and the convolution has 632 segments.

Note that this experiment only studies the size of the convolutions. The discrepancy between the size of the arrangement of the reduced convolutions and that of the complete convolutions is expected to be even larger.

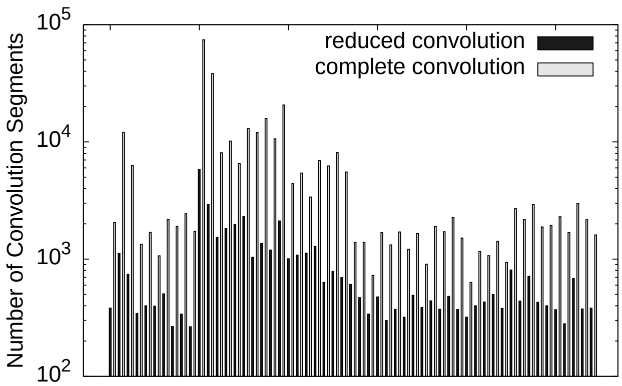


Figure 5: *Number of segments in the reduced convolution and the complete convolution. The size of the reduced convolution is significantly smaller than that of the regular convolution. Note that the y axis is in logarithmic scale.*

6 CONCLUSION

In this paper, we propose a new method based on the idea of the *reduced convolution* for computing the 2-d Minkowski sum of non-convex polygons. We proposed several filters in order to extract the boundaries of the Minkowski sum using the topological properties of the Minkowski sum. The proposed method has been implemented and extensively tested. In Section 5, we showed that the efficiency gain of this proposed approach is indeed from the fact that, in all cases studied, the complexity of the complete convolution is much higher than the complexity of the reduced convolution and that of the final Minkowski sum boundary.

In the near future, we plan to further improve the efficiency and robustness of the proposed method by using exact arithmetic library, such as GMP, and implement better collision detection and segments intersection methods. For the long term goal, the result of this work will be used in providing a more efficient way to compute the $\mathcal{C}$-space mapping. Our preliminary results [Behar and Lien 2010] show that, using the approach developed in this paper, $\mathcal{C}$-space mapping can be made simpler in implementation and often more efficient than the existing techniques.

References

AGARWAL, P. K., FLATO, E., AND HALPERIN, D. 2000. Polygon decomposition for efficient construction of minkowski sums. In *European Symposium on Algorithms*, 20–31.

BAJAJ, C., AND DEY, T. K. 1990. Polygon nesting and robustness. *Inform. Process. Lett. 35*, 23–32.

BARKI, H., DENIS, F., AND DUPONT, F. 2009. Contributing vertices-based minkowski sum of a non-convex polyhedron without fold and a convex polyhedron. In *Shape Modeling and Applications, 2009. SMI 2009. IEEE International Conference on*, 73 –80.

BEHAR, E., AND LIEN, J.-M. 2010. A new method for mapping the configuration space obstacles of polygons. Tech. Rep. GMU-CS-TR-2010-11, George Mason University.

FLATO, E. 2000. *Robuts and Efficient Construction of Planar Minkowski Sums*. M.Sc. thesis, Dept. Comput. Sci., Tel-Aviv Univ., Isael.

FOGEL, E., AND HALPERIN, D. 2006. Exact and efficient construction of Minkowski sums of convex polyhedra with applications. In *Proc. 8th Wrkshp. Alg. Eng. Exper. (Alenex'06)*, 3–15.

FUKUDA, K. 2004. From the zonotope construction to the minkowski addition of convex polytopes. *Journal of Symbolic Computation 38*, 4, 1261–1272.

GHOSH, P. K. 1993. A unified computational framework for Minkowski operations. *Computers and Graphics 17*, 4, 357–378.

GRITZMANN, P., AND STURMFELS, B. 1993. Minkowski addition of polytopes: computational complexity and applications to Gröbner bases. *SIAM J. Discret. Math. 6*, 2, 246–269.

GUIBAS, L. J., AND SEIDEL, R. 1987. Computing convolutions by reciprocal search. *Discrete Comput. Geom. 2*, 175–193.

GUIBAS, L. J., RAMSHAW, L., AND STOLFI, J. 1983. A kinetic framework for computational geometry. In *Proc. 24th Annu. IEEE Sympos. Found. Comput. Sci.*, 100–111.

HALPERIN, D. 2002. Robust geometric computing in motion. *The International Journal of Robotics Research 21*, 3, 219–232.

KAUL, A., AND ROSSIGNAC, J. 1991. Solid-interpolating deformations: construction and animation of PIPs. In *Proc. Eurographics '91*, 493–505.

LIEN, J.-M. 2008. A simple method for computing Minkowski sum boundary in 3d using collision detection. In *The Eighth International Workshop on the Algorithmic Foundations of Robotics (WAFR)*.

LOZANO-PÉREZ, T. 1983. Spatial planning: A configuration space approach. *IEEE Trans. Comput. C-32*, 108–120.

VARADHAN, G., AND MANOCHA, D. 2006. Accurate Minkowski sum approximation of polyhedral models. *Graph. Models 68*, 4, 343–355.

WEIN, R. 2006. Exact and efficient construction of planar Minkowski sums using the convolution method. In *Proc. 14th Annual European Symposium on Algorithms*, 829–840.