Exact Minkowski Sums and Applications*

Eyal Flato Efi Fogel Dan Halperin Eran Leiserowitz

School of Computer Science
Tel Aviv University

ABSTRACT

The *Minkowski sum* of two sets P and Q in $\mathbb{R}^d$ is the set $\{p+q \mid p \in P, q \in Q\}$. Minkowski sums are useful in robot motion planning, computer-aided design and manufacturing (CAD/CAM) and many other areas. In this video we present a software package implemented at Tel Aviv University for the *exact* and efficient construction of Minkowski sums of planar sets. We also explain and demonstrate how Minkowski sums are used in various applications.

Keywords

Minkowski sums, arrangements, motion planning, geometric software, robustness and precision

1. INTRODUCTION

Given two sets P and Q in $\mathbb{R}^d$, their *Minkowski sum* (or vector sum), denoted by $P \oplus Q$, is the set $\{p+q \mid p \in P, q \in Q\}$. Minkowski sums are used in a wide range of applications, including robot motion planning [8], assembly planning [6], and computer-aided design and manufacturing (CAD/CAM) [2].

Consider for example an obstacle P and a robot Q that moves by translation. We can choose a reference point r rigidly attached to Q and suppose that Q is placed such that the reference point coincides with the origin. If we let Q' denote a copy of Q rotated by 180° degrees, then $P \oplus Q'$ is the locus of placements of the point r where $P \cap Q \neq \emptyset$. In the study of motion planning this sum is called a *configuration space obstacle* because Q collides with P when translated along a path π exactly when the point r, moved along π, intersects $P \oplus Q'$.

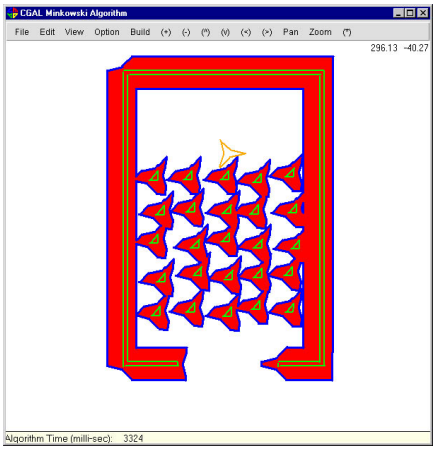

Figure 1: EMINK — screenshot of the interactive program

Much work has been done on obtaining sharp bounds on the size of the Minkowski sum of two sets in two and three dimensions, and on developing fast algorithms for computing Minkowski sums.

In this video we describe a software package, EMINK, developed at Tel Aviv University for the *exact* and efficient construction of Minkowski sums of planar sets—the package is briefly described in the next section. We also explain and demonstrate in the video how Minkowski sums are used in various applications including object placement (arising for example in manufacturing cell layout design) and translational separation (arising in cartography). Figure 1 displays a screenshot of the interactive program.

2. ALGORITHMS AND IMPLEMENTATION

We devised and implemented several algorithms for computing the Minkowski sum of two polygonal sets in $\mathbb{R}^2$ [3]. Our implementation is based on the CGAL software library (www.cgal.org). Our main goal was to produce a robust and exact implementation. This goal was achieved by employing the CGAL *planar maps* and *arrangements* package [4], [7] while using exact number types. We use rational numbers and filtered geometric predicates from LEDA — the Library of Efficient Data-structures and Algorithms [9]. The connection between Minkowski sums and arrangements as well as

*This work has been supported in part by the IST Programme of the EU as a Shared-cost RTD (FET Open) Project under Contract No IST-2000-26473 (ECG - Effective Computational Geometry for Curves and Surfaces), by The Israel Science Foundation founded by the Israel Academy of Sciences and Humanities (Center for Geometric Computing and its Applications), and by the Hermann Minkowski – Minerva Center for Geometry at Tel Aviv University.
Corresponding author: Dan Halperin, danha@tau.ac.il.

SoCG'02, June 5-7, 2002, Barcelona, Spain.
ACM 1-58113-504-1/02/0006.

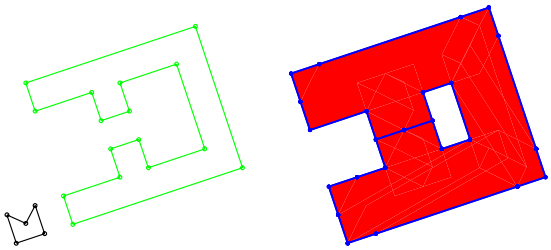

Figure 2: Tight passage: the desired target placement for the small polygon is inside the inner room defined by the larger polygon (left-hand side). In the configuration space (right-hand side) the only possible path to achieve this target passes through the line segment emanating into the hole in the sum.

related implementation projects are described in [5].

We are currently using our software to solve translational motion planning problems in the plane. We are able to compute collision-free paths even in environments cluttered with obstacles, where the robot could only reach a destination placement by moving through tight passages, practically moving in contact with the obstacle boundaries. See Figure 2 for an example. This is in contrast with most existing motion planning software for which tight or narrow passages constitute a significant hurdle.

As mentioned above, beside robot motion planning, we can use Minkowski sums to solve other related problems like minimum distance separation and object placement. Details about these applications are given in [3].

The robustness and exactness of our implementation come at a cost: they slow down the running time of the algorithms in comparison with a more standard implementation that uses floating point arithmetic. This makes it especially necessary to try and speed up the algorithms in other ways. All our algorithms start with decomposing the input polygons into convex subpolygons. We discovered that not only the number of subpolygons in the decomposition of the input polygons but also their shapes had dramatic effect on the running time of the Minkowski-sum algorithms.

We implemented a dozen different methods for decomposing polygons and tested their suitability for efficient construction of Minkowski sums. We implemented various well-known decompositions as well as several new decomposition schemes which are all available in our package. In [1] we report on our experiments with various decompositions and different input polygons. Among our findings are that in general: (i) triangulations are too costly (ii) what constitutes a good decomposition for one of the input polygons depends on the other input polygon — consequently, we develop a procedure for simultaneously decomposing the two polygons such that a "mixed" objective function is minimized, (iii) there are optimal decomposition algorithms that significantly expedite the Minkowski-sum computation, but the decomposition itself is expensive to compute — in such cases simple heuristics that approximate the optimal decomposition perform very well.

Note that convex *covering* (rather than decomposition) is also suitable as a first step in the construction of Minkowski sums. In our work however we focused on convex decomposition.

Recently the package has been extended to construct Minkowski sums of polygons and discs.[1] Since in this case the coordinates of the boundary of the sums need no longer be rational this requires the use of a more elaborate number type: LEDA's "real" [9].

3. EXAMPLES IN THE VIDEO

In the video we focus on how Minkowski sums are used. The examples were produced using our software package. The morphing between the logos of CGAL and LEDA is carried out using the following transformation: if P and Q are two polygons, we compute a sequence of Minkowski sums $tP \oplus (1-t)Q$ for $t \in [0,1]$, where λP is a scaling of the polygon P by factor λ.

4. REFERENCES

[1] P. K. Agarwal, E. Flato, and D. Halperin. Polygon decomposition for efficient construction of Minkowski sums. *Comput. Geom. Theory Appl.*, 21:39–61, 2002.

[2] G. Elber and M.-S. Kim, editors. *Special Issue of Computer Aided Design: Offsets, Sweeps and Minkowski Sums*, volume 31. 1999.

[3] E. Flato. Robust and efficient construction of planar Minkowski sums. Master's thesis, Dept. Comput. Sci., Tel-Aviv Univ., 2000. http://www.cs.tau.ac.il/~flato.

[4] E. Flato, D. Halperin, I. Hanniel, O. Nechushtan, and E. Ezra. The design and implementation of planar maps in CGAL. *The ACM Journal of Experimental Algorithmics*, 5, 2000. Also in LNCS Vol. 1668 (WAE '99), Springer, pp. 154–168.

[5] D. Halperin. Robust geometric computing in motion. In B. Donald, K. Lynch, and D. Rus, editors, *Algorithmic and Computational Robotics: New Dimensions (WAFR '00)*, pages 9–22. A. K. Peters, Wellesley, MA, 2001. To appear in *International Journal of Robotics Research*.

[6] D. Halperin, J.-C. Latombe, and R. H. Wilson. A general framework for assembly planning: The motion space approach. *Algorithmica*, 26:577–601, 2000.

[7] I. Hanniel and D. Halperin. Two-dimensional arrangements in CGAL and adaptive point location for parametric curves. In *Proc. of the 4th Workshop of Algorithm Engineering*, volume 1982 of *Lecture Notes Comput. Sci.*, pages 171–182. Springer-Verlag, 2000.

[8] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1991.

[9] K. Melhorn and S. Näher. *The LEDA Platform of Combinatorial and Geometric Computing*. Cambridge University Press, 1999.

[1] The extension to discs is by Eran Leiserowitz and is based on recently developed tools due to Shai Hirsch and Ron Wein.