

# A Simplex based Dimension Independent Approach for Convex Decomposition of Nonconvex polytopes

Rizwan Bulbul, Farid Karimipour and Andrew U. Frank

Institute of Geoinformation and Cartography  
Technical University of Vienna  
Gusshausstrasse 27-29 E127  
A-1040 Vienna, Austria  
{bulbul, karimipour, frank}@geoinfo.tuwien.ac.at

## 1. Introduction

The problem of decomposing a nonconvex object into its convex components is known as “convex decomposition” and has applications in diverse domains ranging from pattern recognition, motion planning and robotics, computer graphics, image processing and spatial databases, etc. (Ai et al. 2005; Chazelle and Palios 1990; Jianning 2007; Kriegel et al. 1992; Lien and Amato 2004, 2006; Liu et al. 2008). The object representations based on simpler geometric structures are more easily handled (Fernández et al. 2000) than complex structures. The algorithms for convex objects are simple and run faster than for arbitrary objects (Bajaj and Dey 1992; Kriegel et al. 1992; Leonidas 1992; Schachter 1978), for example, the decomposition of complex spatial objects into simple components considerably increases the query processing efficiency (Kriegel et al. 1992). Therefore, the need arises to decompose nonconvex polytopes into simpler convex components.

The majority of the decomposition techniques in literature are dimension dependent and mostly applicable in 2-D and 3-D (Chazelle and Palios 1990) cases only. In this paper, we have proposed a simplex based algorithm for convex decomposition of  $n$ -dimensional polytopes. Unlike current approaches which need separate implementations for each dimension of the input, our approach is a single implementation that works for polytopes of any dimension.

Our approach is based on alternate hierarchical decomposition, AHD (Bulbul and Frank 2009), which recursively decomposes a nonconvex polytope into its components which are convex hulls represented hierarchically in a tree structure called *convex hull tree*, CHT.

The AHD consists of two steps: (1) convex hull computation, and (2) delta region/s extraction (using symmetric difference). The process is then recursively applied to the delta regions until the input region is convex. Suppose that we are given a 2-D nonconvex polytope with a hole (H) in dual space (as shown in fig. 1a) as input. The application of AHD results in the decomposition of the input into its component convex hulls (fig. 1b). The output is a hierarchical representation of convex hulls as CHT (shown in fig. 2). The data structure used for CHT representation is an arbitrary tree, every node of which contains a convex hull. For a dimension independent implementation of AHD we need dimension independent convex hull computation and, delta region extraction.

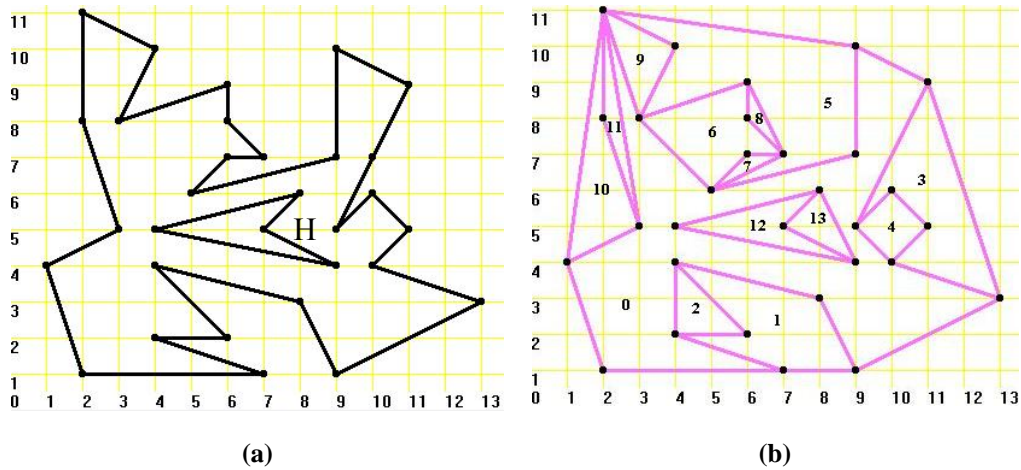


Figure 1: Input polytope with a hole H (a) and its decomposition (b)

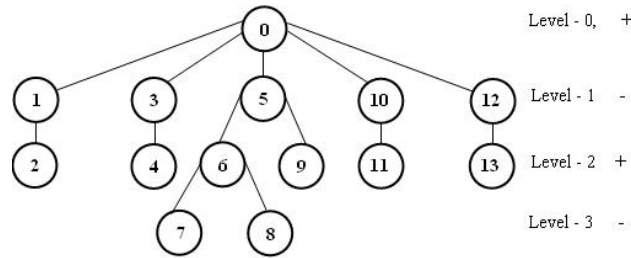


Figure 2: CHT for the example of Fig. 1

There are two possible approaches (fig. 3). First approach deals both of the aforementioned steps independent of each other (fig. 3a). This means any of the existing dimension independent convex hull algorithms (Barber et al. 1996 ; Karimipour et al. 2008) can be used and the result is transformed to the object representation model before extracting the delta regions. In second approach, a data representation model is used that supports both operations to be performed in sequence without conversions (fig. 3b). Our decomposition approach is based on the second approach which is better as it avoids unnecessary conversions.

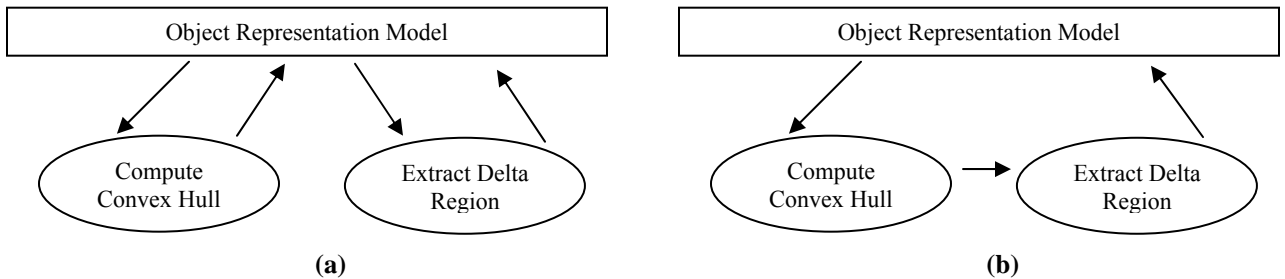
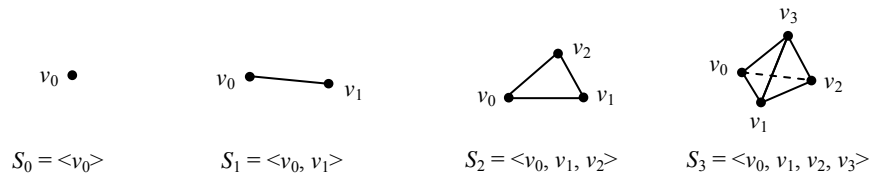


Figure 3: Two possible approaches

## 2. Our Approach: $n$ -Dimensional Implementation of AHD

Our decomposition approach provides a dimension independent implementation of AHD. The approach is based on the simplex based data representation model which provides a consistent implementation of the two basic steps of AHD i.e. convex hull computation and delta region extraction.

A  $n$ -simplex  $S_n$  (abbreviated as *simplex* henceforth) is defined as the smallest convex set in the Euclidean space that contains  $n+1$  of  $n$ -dimensional vertexes  $v_0, \dots, v_n$  that do not lie in a hyperplane of dimension less than  $n$  (Alexandroff 1961). A simplex is represented by the list of its vertexes as  $S_n = \langle v_0 \dots v_n \rangle$ . Fig. 4 shows simplexes of dimensions 0 to 3.



**Figure 4: Simplexes of dimensions 0 to 3: node, edge, triangle, and tetrahedron**

The order of vertexes of a simplex induces an orientation over it:

- The orientation of a 0-simplex (node) is positive.
- The orientation of a 1-simplex (edge) is positive from vertex  $v_0$  to vertex  $v_1$  or negative from vertex  $v_1$  to vertex  $v_0$ .
- The orientation of a 2-simplex (triangles) is defined based on the order in which the vertexes are listed: counter-clockwise (*ccw*) order is positive and clockwise (*cw*) order is negative.
- The orientation of a 3-simplex (tetrahedron) is the sign of the volume constructed by its ordered vertexes.

Orientation of a simplex is changed by odd numbers of permutations of its vertexes; while orientation remains unchanged with even numbers of permutations (Stolfi 1989). For example, for the simplexes of fig.4:

$$S_0 = \langle v_0 \rangle$$

$$S_1 = \langle v_0, v_1 \rangle = - \langle v_0, v_1 \rangle$$

$$S_2 = \langle v_0, v_1, v_2 \rangle = - \langle v_0, v_2, v_1 \rangle = \langle v_2, v_0, v_1 \rangle = \dots$$

$$S_3 = \langle v_0, v_1, v_2, v_3 \rangle = - \langle v_0, v_1, v_3, v_2 \rangle = \langle v_0, v_3, v_1, v_2 \rangle = \dots$$

The boundary of the  $n$ -simplex  $S_n = \langle v_0, \dots, v_n \rangle$  is written as  $\partial S_n$  and is a set of  $n+1$  of  $(n-1)$ -simplexes as follow (Stolfi 1989):

$$\partial S_n = \sum_{i=0}^n (-1)^i \langle v_0, \dots, \bar{v}_i, \dots, v_n \rangle$$

where  $\bar{v}_i$  means omitting the vertex  $v_i$  from the vertex list. For example, for the simplexes of fig.4:

$$\partial S_0 = \varnothing$$


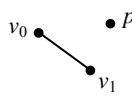
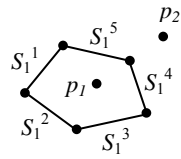
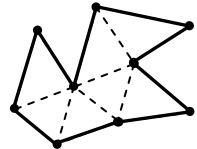
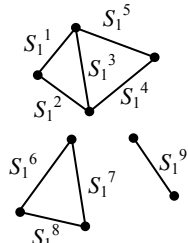
$$\partial S_1 = \langle v_1 \rangle - \langle v_0 \rangle$$

$$\partial S_2 = \langle v_1, v_2 \rangle - \langle v_0, v_2 \rangle + \langle v_0, v_1 \rangle$$

$$\partial S_3 = \langle v_1, v_2, v_3 \rangle - \langle v_0, v_2, v_3 \rangle + \langle v_0, v_1, v_3 \rangle - \langle v_0, v_1, v_2 \rangle$$

Table 1 lists some of the operations on the simplexes, which are used in the next section for developing the decomposition algorithm.

**Table 1: Operations on simplexes**

Operation	Description	Example
$addVertex(v, S)$	Adding a vertex $v$ to a $n$ -simplex $S$ , which produces a $(n+1)$ -simplex	 <p><math>S_1 = \langle v_0, v_1 \rangle</math>      <math>addVertex(v, S_1) = \langle v, v_0, v_1 \rangle</math></p>
$cw(p, S)$ $ccw(p, S)$	Testing the position of a point $p$ respect to a simplex $S$ (clockwise or counter-clockwise)	 <p><math>S_1 = \langle v_0, v_1 \rangle</math>      <math>cw(p, S_1) = false</math> <math>ccw(p, S_1) = true</math></p>
$PtInSimplex(p, \{S_i\})$	Testing if a point $p$ is inside the region delimited by a set of simplexes $\{S_i\}$	 <p><math>R = \{S_1^1, \dots, S_1^5\}</math>      <math>PtInSimplex(p_1, R) = true</math> <math>PtInSimplex(p_2, R) = false</math></p>
$Border(\{S_i\})$	Extracting the border of a set of connected simplexes $\{S_i\}$	 <p><math>R = \{S_i\}</math> <math>(\{S_i\} = \text{all edges})</math>      <math>Border(R) = \{S_j\}</math> <math>(\{S_j\} = \text{bold edges})</math></p>
$SplitRegions(\{S_i\})$	Splitting a set of simplexes $\{S_i\}$ to the connected subsets $\{R_1, \dots, R_k\}$	 <p><math>SplitRegions(\{S_1^1, \dots, S_1^9\}) = (\{S_1^1, \dots, S_1^5\}, \{S_1^6, \dots, S_1^8\}, \{S_1^9\})</math></p>

### 3. Pseudocode: $n$ -Dimensional Implementation of AHD

This section explains the  $n$ -dimensional implementation of a convex hull computation as well as the AHD algorithm for decomposition of polytopes.

### 3.1. Convexhull Computation

The convexhull of a set of points is the smallest region that contains the points. In this paper we use an incremental algorithm called *IncConvexhull* (Berg et al. 2008; Bradford et al. 1996). For some 2D points, the *IncConvexhull* algorithm starts with the triangle goes through the first three points, which is their convexhull. Other points are inserted one by one into the construction and after each insertion, the convexhull is modified: if the inserted point is inside the convexhull, no change is needed in the configuration of the current convexhull; If it is outside, however, its opposite edges are dropped from the convexhull and new edges are added, which are constructed by adding the inserted point to the border points of the dropped edges. Extension of *IncConvexhull* algorithm to  $n$ -dimensional points is possible by using the concept of simplexes. Fig. 5 shows the pseudocode of the  $n$ -dimensional convex hull algorithm.

<p><i>Input.</i> A set of <math>n</math>-dimensional points <math>\{p_0, \dots, p_m\}</math>  <i>Output.</i> A List <math>E</math> contains the <math>(n-1)</math>-simplexes that construct the convexhull of the input points</p> <ol style="list-style-type: none"> <li>1. <math>I \leftarrow</math> an <math>n</math>-simplex constructed by <math>\{p_0, \dots, p_n\}</math></li> <li>2. <math>E \leftarrow</math> boundary of <math>I</math> (which consists of <math>n+1</math> of <math>(n-1)</math>-simplexes)</li> <li>3. <b>for</b> all points <math>p \in \{p_{n+1}, \dots, p_m\}</math></li> <li>4.   <b>if</b> <math>p</math> is outside the region delimited by all <math>(n-1)</math>-simplexes <math>e \in E</math></li> <li>5.     <math>S \leftarrow</math> Set of all <math>(n-1)</math>-simplexes <math>e \in E</math> that that makes a <i>cw</i> turn respect to the point <math>p</math></li> <li>6.     <math>B \leftarrow</math> Set of <math>(n-2)</math>-simplexes that are on the border of <math>S</math></li> <li>7.     <math>N \leftarrow</math> Set of <math>(n-1)</math>-simplexes constructed by adding <math>p</math> to all <math>(n-2)</math>-simplexes <math>b \in B</math></li> <li>8.     <math>E \leftarrow E - S + N</math></li> <li>9. <b>return</b> <math>E</math></li> </ol>
--

Figure 5: Pseudocode of dimension independent incremental convex hull algorithm

### 3.2. $n$ -Dimensional AHD

Using the concept of simplexes, the AHD algorithm is implemented for polytopes of any dimension. The dimension independent convex hull algorithm of fig. 5 and the *splitRegion* function of table 1 are used for a dimension independent AHD, the pseudocode of which is shown in fig. 6.

**Algorithm AHD**

*Input.* A set of  $n$ -simplexes  $R = \{S_0, \dots, S_m\}$  that construct a closed region

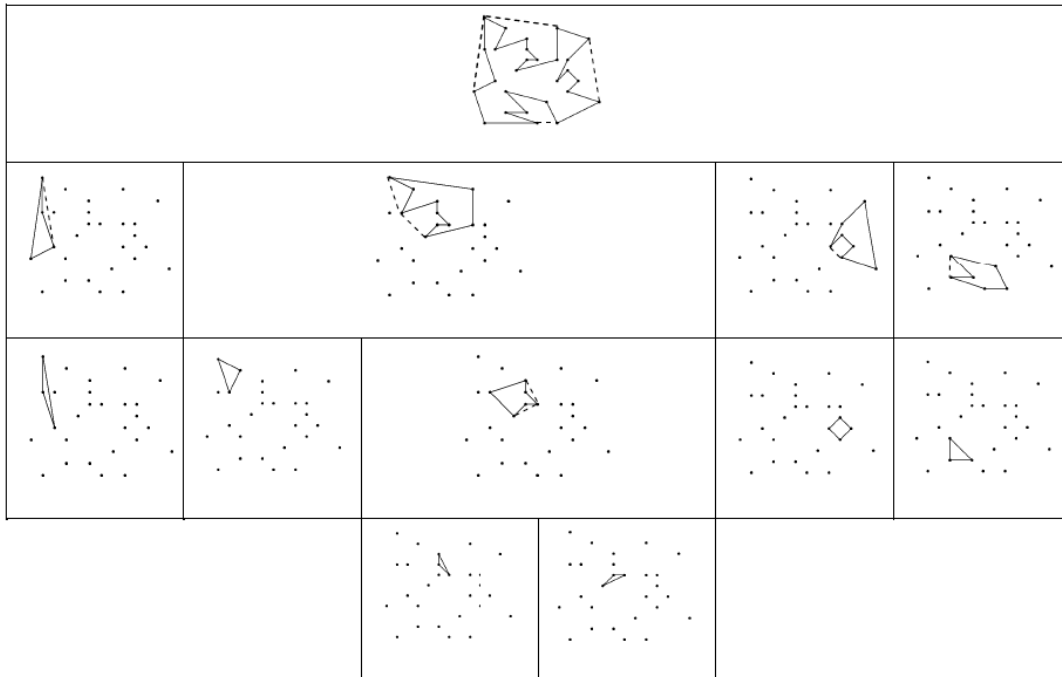
*Output.* A tree of singed convex regions that construct  $R$

1.  $T \leftarrow$  **if**  $sptR$  is empty
2.     **then**  $Node(chR, [])$
3.     **else**  $Node(chR, apply\ DecompositionTree\ on\ all\ elements\ of\ the\ list\ sptR)$
4.     **where**
5.          $sptR$                   $= splitRegions(symD(R, chR))$
6.          $chR$                   $= Convexhull(vs)$
7.          $vs$                   $=$  union of vertexes of input  $n$ -simplexes  $R$
8.          $symD(x, y)$           $= (x - y) + (y - x)$
9.     **return**  $T$

**Figure 6: n-dimensional AHD**

## 4. Implementation

The algorithms have been implemented as Haskell (Jones 2003) modules. The selection of Haskell provides ease of implementation giving simple and compact code. This is possible because of Haskell's built in "list" data structure and its associated higher order functions, lazy evaluation and support for big numbers. Fig. 7 and fig. 8 show the implementation results of our approach for 2D and 3D cases respectively.



**Figure 7: Example - 2D case result**

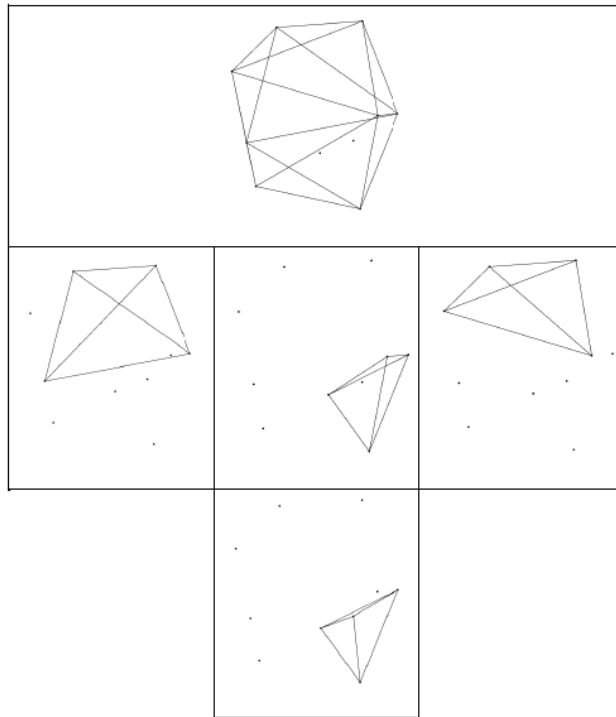


Figure 8: Example- 3D case result

## 5. Conclusion and Future Work

Convex decomposition is the decomposition of nonconvex objects into their convex components and has application is wide range of domains. Most of the current decomposition algorithms are dimension specific needing dimension specific implementations. We provided a simplex based implementation of AHD which decomposes a nonconvex polytope of any dimension into its component convex hulls.

In future, we will apply the decomposition model for the application of Boolean operations (intersection, union and symmetric difference etc.) on nonconvex polytopes of any dimension.

## 6. References

- Ai, Tinghua, Zhilin Li and Yaolin Liu. 2005. "Progressive Transmission of Vector Data Based on Changes Accumulation Model." In *Developments in Spatial Data Handling*.
- Alexandroff, P. 1961. *Elementary Concepts of Topology*. New York, USA: Dover Publications.
- Bajaj, Chanderjit L. and Tamal K. Dey. 1992. "Convex decomposition of polyhedra and robustness." *SIAM Journal on Computing* 21(2):339 - 364
- Barber, C. Bradford, David P. Dobkin and Hannu Huhdanpaa. 1996 "The quickhull algorithm for convex hulls." *ACM Transactions on Mathematical Software (TOMS)* 22(4):469 - 483.
- Berg, M. De, O. Cheong, M. Van-Kreveland and M. Overmars. 2008. *Computational Geometry: Algorithms and Applications (3rd Edition)*: Springer-Verlag.
- Bradford, C., D.P. Dobkin and H. Huhdanpaa. 1996. "The Quickhull Algorithm for Convex Hulls." *ACM Transaction of Mathematical Software* 22(4):469-483.
- Bulbul, Rizwan and Andrew U. Frank. 2009. "AHD: The Alternate Hierarchical Decomposition of Nonconvex Polytopes (Generalization of a Convex Polytope Based Spatial Data Model)." In *17th International Conference on Geoinformatics*. Fairfax, USA.

- Chazelle, Bernard and Leonidas Palios. 1990. "Triangulating a nonconvex polytope." *Discrete and Computational Geometry* 5(1):505-526.
- Fernández, J., L. Cánovas and B. Pelegrín. 2000. "Algorithms for the decomposition of a polygon into convex polygons." *European Journal of Operational Research* 121(2):330-342.
- Jianning, Xu. 2007. "Morphological Decomposition of 2-D Binary Shapes Into Modestly Overlapped Octagonal and Disk Components." *Image Processing, IEEE Transactions on* 16(2):337-348.
- Jones, Simon. 2003. *Haskell 98 Language and Libraries: The Revised Report*. {Cambridge University Press}.
- Karimipour, Farid, Andrew U. Frank and Mahmoud R. Delavar. 2008. "An operation-independent approach to extend 2D spatial operations to 3D and moving objects." In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*. Irvine, California: ACM.
- Kriegel, Hans-Peter, Holger Horn and Michael Schiwietz. 1992. "The performance of object decomposition techniques for spatial query processing." In *Data structures and efficient algorithms*.
- Leonidas, Palios. 1992. "Decomposition problems in computational geometry." In *Department of Computer Sciences: Princeton University*.
- Lien, Jyh-Ming and Nancy M. Amato. 2004. "Approximate convex decomposition." In *Proceedings of the twentieth annual symposium on Computational geometry*. Brooklyn, New York, USA: ACM.
- Lien, Jyh-Ming and Nancy M. Amato. 2006. "Approximate convex decomposition of polygons." *Computational Geometry* 35(1-2):100-123.
- Liu, Rong, Hao Zhang and James Busby. 2008. "Convex hull covering of polygonal scenes for accurate collision detection in games." In *Proceedings of graphics interface 2008*. Windsor, Ontario, Canada: Canadian Information Processing Society.
- Schachter, B. 1978. "Decomposition of Polygons into Convex Sets." *IEEE Computer Society*.
- Stolfi, J. 1989. "Primitives for Computational Geometry." In *Computer Science Department*. Palo Alto, USA: Stanford University.