α-Decomposition of Polygons

Yanyan Lu and Jyh-Ming Lien

George Mason University, Fairfax, Virginia, USA

Mukulika Ghosh and Nancy M. Amato

Texas A&M University, College Station, Texas, USA

Abstract

Decomposing a shape into visually meaningful parts comes naturally to humans, but recreating this fundamental operation in computers has been shown to be difficult. Similar challenges have puzzled researchers in shape reconstruction for decades. In this paper, we recognize the strong connection between shape reconstruction and shape decomposition at a fundamental level and propose a method called α-decomposition. The α-decomposition generates a space of decompositions parametrized by α, the diameter of a circle convolved with the input polygon. As we vary the value of α, some structural features appear and disappear quickly while others persist. Therefore, by analyzing the persistence of the features, we can determine better decompositions that are more robust to both geometrical and topological noise.

1. Introduction

Decomposing a shape into visually meaningful parts comes naturally to humans and researchers believe that decomposition is a fundamental process in shape recognition [17, 18, 21, 7, 8, 3, 36]. However, recreating this fundamental operation in computers has been shown to be difficult [9, 22, 38, 5]. For example, an elbow-like polygon shown in Fig. 1(a) can be decomposed into two subparts in multiple ways, e.g., at \( \overline{pr} \), \( \overline{r_q} \) or any segment in between (see Fig. 1(b)). There are also ambiguities when we consider similar shapes shown in Figs. 1(d) to 1(f). Most existing methods [37, 39, 13, 30, 23, 33, 35] will decompose these polygons into the many small components that provide little information about the structure and shape of the input polygon. Alternatively, we can ignore the holes and simply decompose Figs. 1(c) [1(d) and 1(e)] using \( \overline{pp} \) or \( \overline{r_q} \) in Fig. 1(b). Unfortunately, such decompositions make these three shapes indistinguishable from that in Fig. 1(a) while, structurally, shapes in Figs. 1(a), 1(d) and 1(e) are more similar to each other than to the shapes in Figs. 1(c) and 1(f).

A similar dilemma exists in shape reconstruction. For example, given a point set \( S \), there can be multiple ways to interpret the shape that \( S \) assumes. Much effort has been dedicated to overcome ambiguities [1, 12, 2]. For example, Edelsbrunner et al. [13] proposed α shapes that represent \( S \) as a sequence of shapes parametrized by α, the longest connection allowed between points in \( S \). Edelsbrunner and Mücke [16] gave an interesting intuition for α shapes: given a space populated with stationary beads \( S \), the α shape of \( S \) is simply the subspace that cannot be reached by an α-circle. Therefore, when α goes from zero to infinity, the α shape of \( S \) changes from \( S \) itself to the convex hull of \( S \).
In this paper, we recognize the strong connection between shape reconstruction and shape decomposition at a fundamental level and propose a method called $\alpha$-decomposition. Similar to $\alpha$ shapes, an $\alpha$-decomposition defines a space of decompositions parameterized by the value $\alpha$, the diameter of a circle. The intuition behind our approach is that a polygon can be smoothed by convolving its boundary with a circle. For example, a polygon $P$ gets ‘pufferier’ and insignificant features tend to disappear when it is convolved with a larger circle. In particular, as the value of $\alpha$ goes from zero to infinity, the $\alpha$-decompositions of $P$ form a space ranging from an exact convex decomposition of $P$ to $P$ itself (i.e., no decomposition). Therefore, $\alpha$-decomposition provides a unified way to distinguish (with small $\alpha$) and to categorize (with large $\alpha$) the polygons in Fig. 1.

Additionally, a space of decompositions (in contrast to a single decomposition) provides a critical benefit that allows persistence analysis. When we vary the value of $\alpha$, some structural features (in both shape and $\alpha$-decomposition) appear and disappear quickly as others persist. Therefore, by analyzing the persistence (i.e., life span) of the features, we can determine better shape decompositions that are more robust to both geometrical and topological noise.

**Main Contribution.** In this paper, we provide a formal definition of $\alpha$-decomposition based on convolution. We show that the vertices in the arrangement of convolution are closely related to well-known features, such as bridges and pockets, related to the concavity of polygons. By varying the value of $\alpha$, these concave features form a hierarchy, that naturally encodes the persistence (i.e., life span) of concave features in terms of their concavity measurement. Our results using the MPEG 7 image set show a direct connection between the concavity measurement and the significance of these shape features. Finally, diagonals connecting concave features are used to cut (decompose) the input polygon. Cut selection in an $\alpha$-decomposition is a constrained optimization problem that maximizes the total score of the selected cuts and is formulated as a multiply constrained knapsack problem. We show experimentally that $\alpha$-decomposition consistently produces natural decompositions for a variety of shapes in the MPEG 7 image set and does so without unintuitive parameter tuning required by other methods.

The main benefits of $\alpha$-decomposition can be summarized in Fig. 2. The (red) dots in Fig. 2 are detected concave features (pocket minima) and the size of the dots indicates the relative significance of the features. In Fig. 2 and in the rest of this paper, we define the value of $\alpha$ with respect to the diameter of the minimum enclosing circle of the input polygon. Fig. 2(a) shows the decomposition with a small value $\alpha = 0.05$ (i.e., 0.05 of the diameter of the minimum enclosing circle of the chicken polygon) in which detailed features are identified while some important structural features (e.g., the concave feature at the back of the chicken) are missing. On the other hand, when $\alpha = 1$, Fig. 2(b) shows that only features with high concavity are detected, and detailed features that can potentially help to produce more natural decompositions are missing. Finally, in Fig. 2(c) persistence analysis is applied to consider features detected for $0.05 \leq \alpha \leq 1$.

Figs. 2(d) and 2(e) show the difference between the results from $\alpha$-decomposition and approximate convex decomposition (ACD) for polygons with holes. By default, $\alpha$-decomposition ignores these small holes since they either disappear after convolution or their concavity is not significant enough. On the other hand, ACD, and most existing methods [17, 39, 13, 30, 23, 33, 35], consider these holes (and other types of concave features shown in Fig. 1) as concave features that must be resolved.

As is discussed later, we note that $\alpha$-decomposition also
has strong connection to the medial axis (MA), an important shape descriptor. Our approach to create $\alpha$-decompositions relies heavily on detecting the intersections of shape convolution. These intersections parameterized by $\alpha$ implicitly trace out the MA in the space exterior to the polygon $P$. Thus, the persistence analysis mentioned earlier corresponds to measuring the length of the segments on the MA. However, $\alpha$-decomposition has significant differences from existing MA-based decomposition methods, such as $[37, 39]$, that usually consider the MA interior to the polygon. A more detailed comparison between $\alpha$-decomposition and the MA-based decomposition methods can be found in Section 6.1.

**Limitations.** Although our results show that our current implementation of $\alpha$-decomposition has reasonable running times (a few seconds) for polygons with thousands of vertices, its efficiency can still be significantly improved by considering the coherence in the combinatorial structure of the convolution between different $\alpha$ values. The second major limitation is the lack of well-defined criteria for quality comparisons. This limitation is not specific to $\alpha$-decomposition, and we also note that this limitation is further hindered by the lack of public domain implementations of many existing methods. Our implementation of $\alpha$-decomposition and the results will be released to stimulate future research (visit http://masc.cs.gmu.edu for more details).

2. Related Work

Polygon decomposition has been extensive studied theoretically and experimentally in areas including computer graphics, computational geometry [26], computer vision and pattern recognition. Studies in computer graphics focus mainly on 3-d models. Interested readers can refer to recent works on shape partitioning [24, 29, 42] and the reviews therein.

In computational geometry, researchers are traditionally interested in creating decompositions subject to some optimization criteria, such as a minimum number of convex components [10, 19, 25, 11]. Most of these problems are shown to be in NP-hard [31, 25]. More recently, several methods have been proposed to partition at salient features of a polygon. Simmons and Séquin [37] proposed a decomposition using an **axial shape graph**, a weighted medial axis. Tănase and Veltkamp [39] decompose a polygon based on the events that occur during the construction of a straight-line skeleton. Dey et al. [13] partition a polygon into **stable manifolds** which are collections of Delaunay triangles of sampled points on the polygon boundary. Lien and Amato [40] partition a polygon into approximately convex components. Their method reveals significant shape structures by recursively resolving the most concave features until the concavity of every component is below some user-specified threshold. Wan [40] extends [40] to incorporate both concavity and curvatures and prevent over segmentation by avoiding cuts inside pockets.

In pattern recognition and computer vision, shape decomposition is usually a step toward shape recognition. For instance, Siddiqi and Kimia [36] use curvature and region information to identify limbs and necks of a polygon and use them to perform decomposition. Recently, Liu et al. [33] and Ren et al. [35] have proposed to create fewer and more natural nearly convex shapes. Both methods [33, 35] use mutex pairs to enforce the concavity constraint. Points $p_1$ and $p_2$ form a mutex pair if their straight line connection is not completely inside the given shape. Their focus is on separating all mutex pairs whose concavity-based weights are larger than a user-specified threshold. Liu et al. [33] used linear programming to compute decomposition with minimum cost, and Ren et al. [35] applied a dynamic subgradient-based branch-and-bound search strategy to get minimum number of cuts. Similarly, Juengling and Mitchell [23] formulate decomposition of a polygon as an optimization problem and apply dynamic programming to find the optimal subset of cuts from all possible cuts. The objective functions used for optimization favors short cuts that create dihedral angles close to $\pi$. Mi and DeCarlo [34] propose to decompose shape into elliptical regions glued by a hyperbolic patches. Their method defines the idea of relatability based on smoothed local symmetries that measure how easily two separate curves can be joined together smoothly and naturally. Thus, reasonable cuts are placed at places where relatability increases quickly.

An important requirement in shape decomposition is its robustness to boundary noise. Several of these methods require pre-processing (e.g., model simplification [23, 39]) or post-processing (e.g., merging overpartitioned components [13, 54]) due to boundary noise. Other methods [30, 33, 35] are designed to tolerate these noise. However, as far as we know, no existing approaches focused on handling topological noise that appear quite commonly in polygons generated from images. $\alpha$-decomposition is also unique in the way that it creates and analyzes the decomposition space instead of a single decomposition.
3. Preliminaries

The input of α-decomposition is a polygon $P$ represented by a set of $n$ disjoint boundaries $\{P_0, P_1, \ldots, P_{n-1}\}$, where $P_0$ is the external boundary and $P_{k>0}$ are boundaries of holes. Each boundary consists of an ordered set of vertices $\{p_i\}$ which defines a set of edges each of which starts at vertex $p_i$ as $e_i = p_ip_{i+1}$, and each edge $e_i$ has an outward normal $\vec{n}_i$. A polygonal boundary is simple if no nonadjacent edges intersect. Thus, a polygon $P$ with nested simple boundaries is the region enclosed in $P_0$ minus the region enclosed in $\cup_{k>0} P_k$.

The implementation of α-decomposition depends heavily on the concept of convolution. The convolution of two polygons $P$ and $Q$, denoted as $P \odot Q$, is a set of line segments generated by “combining” the segments of $P$ and $Q$ [20]. In this paper, $Q$ is a circle with diameter $\alpha$. To simplify our discussion, we will call such circle an α-circle. The convolution of polygon $P$ and an α-circle is called α-convolution and will be the basis for determining bridges, pockets and concavities of $P$. The α-convolution is composed of (1) edges of $P$ translated by $\alpha/2$ in the outward normal direction of the edge and (2) arcs with radius $\alpha$ centered at the vertices of $P$ connecting the end points of the translated edges. In the next section, we will discuss how the α-convolution can be used to measure concavity.

4. α-Concavity

Intuitively, concavity is a measure of depth in a pocket-like portion of the polygon, and α-concavity is simply the concavity measured from the α-convolution. In this section, we will first provide formal definitions of pocket, bridge, and concavity (in Section 4.1.1). Then, we will show how these features can be determined from α-convolution (in Section 4.2). Finally, we will discuss the relationships between the concavities at a given $\alpha$ and between different $\alpha$ values (in Section 4.3).

4.1. Bridges, Pockets and Concavity

**Definition 4.1.** A bridge $\beta$ of a given polygon $P$ is a segment $\beta = vu$ that lies completely in the space exterior to $P$, where $v$ and $u$ are two points on the boundary $\partial P$ of $P$. More specifically, a segment $\overline{vu}$ is a bridge of $P$ if and only if $v, u \in \partial P$ and the open set of $\overline{vu}$ is a subset of the complement of $P$.

![Figure 3: Bridges $\beta_0, \beta_1, \beta_2$, and $\beta_3$, and their pocket minima. Bridges $\beta_0$ and $\beta_1$ are the children of $\beta_2$.](image)

Therefore, a bridge cannot enter $P$ or intersect the boundary of $P$ except at the end points. Examples of bridge are shown in Fig. 3. Note that, this definition of bridge is more general than that in [30] where a bridge must be on the convex hull of $P$ (e.g. $\beta_2$ in Fig. 3).

**Definition 4.2.** A pocket $\rho$ is associated with a bridge $\beta = vu$ and is an interval of a boundary (either external or hole of $P$) between $v$ and $u$ so that the region enclosed by $\beta$ and $\rho$ is in $P$, the complement of $P$.

Intuitively, when traversing the boundary of $P$, a bridge can be viewed as a short cut over its pocket. For example, the pocket of the bridge $\beta_0$ in Fig. 3 is a polyline between vertices $d$ and $e$ via $x$. Note that, even though we do not restrict the bridge to be a convex hull edge, the pocket must not be part of convex hull of $P$. This property is proved in the following lemma.

**Lemma 4.3.** A pocket $\rho$ of a polygon $P$ must be in $\partial P \setminus \partial \text{CH}(P)$, where $\partial \text{CH}(P)$ is the boundary of the convex hull of $P$.

**Proof.** By definition, the area enclosed by $\rho$ and its associated bridge $\beta$ must be a negative area. If $\rho$ is on convex hull boundary $\partial \text{CH}(P)$ then $\rho$ must coincide with $\beta$ and the enclosed area must be zero. This contradicts to the definition of $\rho$. \hfill $\Box$

In α-decomposition, bridges and pockets form a hierarchy. For example, in Fig. 3 bridges $\beta_0$ and $\beta_1$ are both children of $\beta_2$. Details on how this hierarchy can be formed (either at a given $\alpha$ or at different $\alpha$ values) will be discussed in Section 4.3. The measure of how concave a pocket $\rho$ is depends on $\rho$‘s position in the hierarchy. Intuitively, the distance from a vertex in the pocket to its associated bridge provides an important measure. Therefore, we define concavity as:

**Definition 4.4.** For a pocket $\rho$ without children, we define the concavity $\rho$ as the longest distance from a vertex in $\rho$ to the bridge $\beta$. More specifically,

$$\text{concavity}(\rho) = \max_{v \in \rho} (\text{dist}(v, \beta))$$

where $\text{dist}(v, \beta)$ is the distance between vertex $v$ and $\beta$. 

For a pocket $\rho$ with children $\mathcal{R}$, the concavity is

$$\max_{\rho' \in \mathcal{R}} \left( \text{concavity}(\rho') + \text{dist}(\beta', \beta) \right)$$

where $\beta'$ is the bridge of $\rho'$ and dist($\beta', \beta$) is the distance between bridges $\beta'$ and $\beta$. In both cases, the vertex in $\rho$ that realizes concavity($\rho$) is called [pocket minimum](#).

In Fig. 3 bridges $\beta_0$ and $\beta_2$ have pocket minimum $x$ and $\beta_1$ has pocket minimum $y$. Note that even though in our discussion we will assume that the straight-line distance is used, dist($v, \beta$) and dist($\beta', \beta$) can be measured through more sophisticated distance metrics, such as shortest-path distance. In fact, measuring concavity using straight-line distance with the hierarchy closely approximates the concavity measured solely by shortest-path distance without the hierarchy.

4.2. $\alpha$-Convolution and Bridge

There exists a strong connection between the $\alpha$-convolution, the medial axis and bridges. In particular, if an intersection $x$ in the $\alpha$-convolution is on the medial axis in the space exterior to $P$, then a bridge can always be created near $x$ as illustrated in Fig. 4. Lemma 4.5 proves this claim.

**Lemma 4.5.** An intersection $x$ of two non-adjacent boundary elements of the $\alpha$-convolution can create a bridge if and only if $x$ is on the medial axis of $\overline{P}$, the complement of polygon $P$.

**Proof.** Let $x$ be an intersection between two non-adjacent boundary elements of the convolution. Because the boundary elements can only be either a line segment from an edge of $P$ or an arc centered at a vertex of $P$, $x$ can be directly mapped back to $P$. We call these edges and vertices the source pair of $x$. The source pair of $x$ can be two vertices, two edges or a pair of vertex and edge. If we can find a segment connecting the source pair without intersecting the boundary of $P$, then this segment is a bridge. To ensure this, we can place the $\alpha$-circle at $x$ and if the $\alpha$-circle is empty of $P$, then we can always make the connection between source pair of $x$. In fact the $\alpha$-circle must touch $P$ and the source pair, thus the bridge is simply the connection between the tangent points between the $\alpha$-circle and the source pair.

**Definition 4.6.** An $\alpha$-bridge is a line segment between the tangent points of an empty $\alpha$-circle centered at an intersection $x$ of the $\alpha$-convolution.

Finally, we simply define an $\alpha$-concavity as the concavity in a pocket associated with an $\alpha$-bridge. An interesting property with the $\alpha$-bridge and $\alpha$-concavity is that when $\alpha$ approaches zero, every reflex vertex in $P$ is an $\alpha$-concavity and its $\alpha$-bridge is the segment connecting the incident edge of the reflex vertex. When $\alpha$ approaches $\infty$, $\alpha$-bridge becomes an edge on the convex hull and $\alpha$-concavity becomes the vertex furthest away from the boundary of the convex hull.

4.2.1. Determine All $\alpha$-Bridges in $P$

Despite the connection between convolution and bridge, not every intersection in the convolution forms an $\alpha$-bridge. A $\alpha$-bridge is formed by an $\alpha$-convolution intersection $v$ when an $\alpha$-circle centered at $v$ is empty. To determine all $\alpha$-bridges for a given $\alpha$, a naïve approach checks if the $\alpha$-circle is empty at all intersections. A more efficient approach based on the geometric properties shown in Lemma 4.7 reduces the number of such checks significantly.

**Lemma 4.7.** When an intersection on an orientable loop $L$ of the convolution forms a bridge then all other intersections on $L$ form bridges. Otherwise, no bridges can be formed for $L$.

**Proof.** A loop is orientable if all the normal directions of the edges in the loop are all either pointing inward or outward. Moreover, if the $\alpha$-circle is empty at intersection $x$ then $x$ must be a point on the boundary of the Minkowski sum of $P$ and the $\alpha$-circle. Let $A$ be the arrangement of the segments and arcs in the convolution. Let $\ell$ be a loop extracted. It is guaranteed that $\ell$ must be empty since we trace $\ell$ by making the largest right turns at every intersection. Since $\ell$ is empty, we know that $\ell \subset A$. Furthermore, since we know that all vertices in each cell of $A$ must have the same winding number. Therefore, we know that all points on $\ell$ will have the same winding number. If $\ell$ is not a Minkowski sum boundary, then all intersection on $\ell$ will have positive winding numbers.
More specifically, α-decomposition only performs a single intersection check on each orientable loop, thus reduces the number of checks from $O(n^2)$ to $O(n)$, where $n$ and $n_f$ are the number of vertices and folds, respectively, in the input polygon $P$. A fold $f$ is a sub-polygonal chain of $\partial P$ that has accumulated right turn larger than $\pi$. For example, the only fold in the Fig. 5 is the pocket between the beetle’s legs. By definition, holes are also folds, and folds are the only sources that can form loops in the convolution.

Proof. We provide a simple prove sketch here. As we mentioned earlier, when $\alpha$ increases, less significant features disappear (e.g. Fig. 4(b)), therefore, the longer a concave feature can survive, the more significant it is. This property is nicely encoded in the hierarchy. Recall that the pocket minimum of a pocket $p$ with children is the deepest pocket minimum $m$ of $p$’s children measured from $p$’s bridge (Definition 4.4). Therefore, we say $m$ survives in $p$ while the pocket minima in other kids pockets die. When a pocket minimum dies, its concavity measurement will not be updated even when $\alpha$ increases. Moreover, when a pocket minimum continues to survive with increasing $\alpha$, its concavity must increase monotonically.

**Lemma 4.9.** When the value of $\alpha$ increases, the concavity of surviving pocket minimum must also increase. **Proof.** We provide a simple prove sketch here. As we mentioned earlier, when $\alpha$ increases, the bridges converge to convex hull edges. Therefore an $\alpha$-bridge can only move “away” from the previous bridge and pocket thus increases the concavity.

As a consequence, the persistence of a concave feature is nicely encoded by the concavity measured before the feature dies.

5. $\alpha$-Decomposition

After the pocket minima of a given polygon $P$ are ordered by their concavities and organized into a hierarchy, the next step is to determine the cuts connecting these pocket minima. In our approach, we first compute a set of potential cuts using the diagonals in the constrained Delaunay triangulation (CDT) of simplified $P$. Details are discussed in Section 5.1. For each pocket...
minimum, the potential cuts are grouped into cut sets such that the cuts in each cut set must be selected together in order to resolve a pocket minimum. We will discuss how a cut set is defined in Section 5.2. Finally, to create the α-decomposition, we determine the cuts that resolve all intolerable pocket minima while maximize the total scores subject to the constraints that no conflicting cut sets are selected. We address this constrained optimization problem by solving the multiply constrained knapsack optimization. Details will be discussed in Section 5.3.

5.1. Potential Cuts, Cut Evaluation and Early Rejection

It has been shown that the diagonals in Constrained Delaunay Triangulation (CDT) contain cuts to create a natural looking decomposition since Delaunay Triangulation tend to avoid skinny components [23]. Given a polygon $P$, the potential cuts of $P$ are the diagonals of the CDT of the simplified polygon $\tilde{P}$ of $P$. The simplified polygon $\tilde{P}$ is composed of pocket minima and the vertices between every two consecutive bridges. Essentially, $\tilde{P}$ is $P$ with all pocket vertices replaced by the pocket minima. Fig. 6 shows the simplified polygons for various α values.

In order to prepare for the final cut selection, each potential cut will be evaluated, and then grouped into cut sets. To evaluate a potential cut $\delta$, we consider the concavities of its two end vertices $s$ and $t$ and its length $l$, i.e., the score of $\delta$ is simply:

$$V(\delta) = (\text{concavity}(s) + \text{concavity}(t))/l.$$  \hspace{1cm} (3)

This score function is similar to those in [30][33][35][23].

Depends on the complexity of the (simplified) shapes, there can be an excessive number of diagonals. Many of these diagonals could be filtered out before we perform more expensive analysis to create the final cuts. Our idea for rejecting these potential cuts is simple. We want to keep enough potential cuts so that all pocket minima can still be resolved.

For example, when a diagonal connects vertices outside the pockets or connects pocket minima in the same subtree of the hierarchy, the diagonal can be simply rejected as it does not resolve any concavity. Given a pocket minimum $v$, a diagonal connecting $v$ to a vertex outside pockets can be removed if it cannot subdivide the dihedral angles of $v$ to angles smaller than $\pi$. Even for those that can resolve $v$, we can simply keep one diagonal that has the highest score as a backup cut when $v$ cannot be connected to other pocket minima. According to our experiments in Section 6, these early rejection steps reduce the number of potential cuts by 73%.

5.2. Minimum Cut Set and Conflicting Cut Sets

After we identify a set of potential cuts, diagonals incident to each vertex $v$ are grouped into minimum cut sets. A cut set contains a set of potential cuts that need to be resolved together so that the dihedral angles of $v$ are smaller than $\pi$. A cut set is said to be minimum if no diagonals can be removed from the set without violation the definition of cut set. It is not difficult to show that each minimum cut set will have at most two potential cuts. Each diagonal can belong to multiple cut set, and each vertex $v$ can have multiple cut sets. An example of cut sets for vertices $y$ and $x$ is shown in Fig. 6.

The cut sets from a vertex $v$ are mutually exclusive because a single cut set, by definition, can resolve $v$. As a consequence, no two cut sets should be selected for $v$ in the final decomposition. To ease our discussion, we define the complement $C_i$ of a cut set $C_i$ be a set of conflicting cuts sets. For example, the complement of cut set $\{\overline{vy}, \overline{yv}\}$ (for vertex $v$) is the cut set $\{\overline{vx}, \overline{vx}\}$. Note that the conflicting relationship is for the cut sets; not for individual diagonals. For example, cut set $\{\overline{vx}\}$ conflicts with $\{\overline{xa}, \overline{xh}\}$ and $\{\overline{xc}, \overline{xb}\}$ but not with $\{\overline{vy}\}$ since $\{\overline{vy}\}$ is not at a cut set of vertex $v$. A cut set can also conflict with cut sets from multiple (at most three) vertices. The complement of the cut set $\{\overline{vx}, \overline{vx}\}$ includes cut sets from vertices $x$ and $v$, i.e., $\{\overline{vy}\}, \{\overline{xa}, \overline{xh}\}$ and $\{\overline{xc}, \overline{xb}\}$.

5.3. Cut Selection as Multiply Constrained Knapsack

Cut selection for α-decomposition is a constrained optimization problem that resolves all intolerable pocket minimum and maximizes the total score of the selected cuts without including conflicting cut sets.

Let $C$ be a set of potential cuts incident to all pocket minima whose concavity is larger than $\tau$, a user specified concavity tolerance. Let $C_i \subset C$ be a cut set (i.e.,
diagonals incident to a vertex that can be resolved together). Recall that $C_i$ is the conflicting cut sets of $C_i$. To formally define our problem, we let $|C_i|$ be the number of diagonals in $C_i$ and let $V(C_i)$ be the total score of all the diagonals in the cut set, i.e., $V(C_i) = \sum_{\delta \in C_i} V(\delta)$. Our goal is to select a subset $\mathcal{K}$ of all cut sets so that $V(\bigcup_{C_i \in \mathcal{K}} C_i)$ is maximized subject to the constraints (1) $|\bigcup_{C_i \in \mathcal{K}} C_i| \leq n$ and (2) $\mathcal{K} \cap C_i$ is empty for all $C_i \in \mathcal{K}$.

We observe that the above description can be formulated as multiply constrained knapsack problem. Let the optimal solution $S(n,m)$ of the optimal solution $S(n,m)$ can be represented recursively as:

$$V(n,m) = \max_{0 \leq k \leq \alpha} \begin{cases} V(n,m-1), & f(n-k,m-1) + f \cdot \delta \\ V(n-k,m-1) + f \cdot \delta, & \text{otherwise} \end{cases}$$

(4)

where $f$ is an indicator function defined as:

$$f = \begin{cases} 0 & \left| \left( S(n-k,m-1) \setminus \overline{C_m} \right) \cup C_m \right| > n \\ 1 & \text{otherwise} \end{cases}$$

(5)

and $\delta$ is the score difference after the cut set $D_m$ is added to the final cuts:

$$\delta = V(C_m \setminus S(n-k,m-1)) - V(S(n-k,m-1) \cap \overline{C_m})$$

The indicator function $f$ has value zero if we cannot add the cut set $C_m$ to the optimal solution $S(n-k,m-1)$.

This optimization problem can be solved using the classic memorization method in dynamic programming. Since we only consider minimum cut set and the size $|C_m|$ of each minimum cut set $C_m$ is at most 2, the time complexity for solving Eqn. 4 takes only constant time. Note that, because all cut sets in $S(n,m)$ must be conflict free, we may have to visit $S(n,m)$ multiple time to make sure $V(n,m)$ stabilizes.

6. Results and Discussion

We implemented $\alpha$-decomposition in C++. The experimental results shown in this section are collected on a laptop with Intel Core 2 Duo at 2.53 GHz and 8 GB memory. Polygons converted from MPEG 7 image set [27] are used in our experiments. As a result, the polygon boundaries are not smooth and, in some examples, the polygon is in fact not simple and includes self-intersections. In general, the edge length in each of these polygons is around the pixel length of the input images. Some sample results are shown in Figs. 7 to 15, and we will provide detailed discussion in the rest of this section. More example output and the polygons created from MPEG 7 image set can be found at our project page at http://masc.cs.gmu.edu.

It is important to recall that we define the value of $\alpha$ with respect to the diameter of the minimum enclosing circle of the input polygon. Therefore, when $\alpha = 0.5$, it means the $\alpha$-circle used for $\alpha$-decomposition is half the size of the minimum enclosing circle of the input polygon.

6.1. Advantages of $\alpha$-decomposition

$\alpha$-decomposition encodes persistence of concave features. As we have already seen in Fig. 2 concave features can be revealed at different $\alpha$ values and the concavity hierarchy naturally encodes the persistence of these concave features. Fig. 7 provides another example to support this observation. In particular, the concave features on the back of the deer’s neck are revealed at $\alpha = 0.2$ and $\alpha = 0.5$.

$\alpha$-decomposition is robust to topological noise. $\alpha$-decomposition provides a straightforward way to handle topological noise. As shown in Fig. 2 and Fig. 8 when random holes are added to the polygons, $\alpha$-decomposition can still produce meaningful decompositions. We would like to point out that, although only circular holes are used to demonstrate this benefit, any
α-decomposition produces natural decompositions of deformed and transformed shapes. We further study the robustness of α-decomposition in five categories of MPEG 7 dataset. Figs. [11] to [15] show the α-decompositions, and Table [1] shows the running times. These results are again created with the same concavity tolerance and α value range as mentioned above. In each dataset, similar shapes are deformed and transformed in several ways. For example, in octopus dataset, their arms have different length and the bodies have different roundness. Figs. [12] and [13] show dogs and horses with various body types. Figs. [14] and [15] show results of two types of insects: fly and beetle, which are two most complicated sets of shapes in MPEG 7 dataset. We found that the α-decompositions of these models are quite natural given that the input parameters are fixed.

α-decomposition produces natural decompositions without parameter tuning. Finally, we would like to point out that α-decomposition does not require unintuitive parameters that are usually necessary for many other shape decomposition methods, e.g. weights in objective function [30, 23, 35] and number of mutex pairs [33, 35]. In ACD, each cut is scored using weighted cut length and weighted concavity. The dog and octopus polygons in Fig. [10] are decomposed using different weight combinations. The ability to encode persistence analysis in its concavity is the main reason that α-decomposition can avoid these unintuitive parameters. In this paper, we consistently use 0.05 ≤ α ≤ 1. Usually, the upper bound (in our case α = 1) is not much a concern since most features are removed even when the shape is convolved with its own bounding circle. The lower bound in general should be a very small value (in our case α = 0.05). The lower bound can be a value very close zero, but, after the persistence analysis, many of these concave features revealed by small alpha-values are ignored.

α-decomposition vs. ACD. To further justify the benefits of α-decomposition, we compare our results to the approximate convex decomposition (ACD) [30]. The ACDs of octopus and dogs (from the first six polygons in the data set) are shown in Fig. [10]. Due to the natural of greediness, ACD identifies the most concave feature and resolves the feature without considering other concave features. This becomes a serious problem for the octopus polygons. To disjoint an arm from the body of an octopus, we need to connect two consecutive concave features along its boundary. However, when the “coordination” between concave features is not critical, ACD does generate natural looking decompositions. For example, the dog polygons in Fig. [10] are nicely decomposed by ACD even though the decompositions are quite different from those generated by α-decomposition (shown in Fig. [12]).
α-decomposition vs. the MA-based methods. α-decomposition has close relationship to the medial axis (MA). However, α-decomposition has several advantages over the existing MA-based approaches that do not distinguish the holes that are topological noise and the holes that have significant contribution to the overall structure of the shape. Moreover, unlike most MA-based methods that usually require some filtering process to remove insignificant branches or weight the segments in MA [37, 39], α-decomposition tracks the intersections in α-convolution outside the shape and these intersections always rapidly stabilize to a few representative intersections when α increases.

6.2. Limitations

Although Table [1] shows that our current implementation of α-decomposition has reasonable running times (a few seconds) for polygons with thousands of vertices, its efficiency can still be significantly improved. This is particularly true for the first step that identifies the intersections in convolution when multiple α values are considered. For example, we can consider the coherence in the combinatorial structure of the convolution and its arrangement between different α values. It has been shown in our previous research [6] that considering only critical changes without rebuilding the convolution from scratch can provide speedup in several orders of magnitude. Therefore, we believe that it is highly possible to reduce the running time to under a second.

The second major limitation is the lack of well-defined criteria for quality comparisons. As in many previous works, our results shown in this sections are evaluated visually. Although several benchmarks do exist, all comparative studies in previous works focus on quantitative differences, e.g., number of components in the final decompositions [30, 33, 35]. Therefore, this limitation is not specific to α-decomposition, and the answer to address this issue may rely on psychology-based empirical methods. However, we also realize that this limitation is further hindered by the lack of public domain implementation of many existing methods. Similar to ACD [30], our implementation of α-decomposition and the results shown in this section will be released to stimulate future research.

7. Conclusion and Future Work

In this paper, we propose a decomposition method called α-decomposition. Our method encodes persistence analysis for concave features in concavity measurement. To select optimal cuts, α-decomposition performs a constrained optimization strategy to maximize the total score. As a result, the new method produces more meaningful components compared to ACD [30].
Figure 10: Results from ACD [30] of the first rows of octopus and dog datasets. The concavity tolerance is 0.05. The weight parameters for concavity and cut length are both set to 0.5 (i.e. equally important).

Future Works. As mentioned in Section 6.2, convolution is computed from scratch when $\alpha$ changes. In the future, we are interested in continuously updating the convolution for multiple $\alpha$ values. Additionally, $\alpha$-decomposition can be easily parallelized over $\alpha$ to gain more speedup. This continuous update and parallelization will be more desirable when $\alpha$-decomposition is extended to 3D models.

Acknowledgment

This work of Lien and Lu is supported in part by NSF IIS-096053, Autodesk and FHWA. The work of Amato and Ghosh is supported in part by NSF awards CRI-0551685, CCF-0833199, CCF-0830753, IIS-096053, IIS-0917266, by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST). We also thank the anonymous reviewers for the constructive comments.


Figure 11: MPEG 7 octopus. The average size of the octopus polygons is 1321. In all decompositions, the $\alpha$ values used in persistence analysis are between 0.05 and 1. The concavity tolerance $\tau$ is 0.05. The average size of the decomposition is 9.

Figure 12: MPEG 7 dogs. The average size of the dog polygons is 1250. In all decompositions, the $\alpha$ values used in persistence analysis are between 0.05 and 1. The concavity tolerance $\tau$ is 0.05. The average size of the decomposition is 8 (max 10 and min 7).

Figure 13: MPEG 7 horse. The average size of the horse polygons is 2347. In all decompositions, the $\alpha$ values used in persistence analysis are between 0.05 and 1. The concavity tolerance $\tau$ is 0.05. The average size of the decomposition is 8 (max 10 and min 7).
Figure 14: MPEG 7 fly. The average size of the fly polygons is 1662. In all decompositions, the \( \alpha \) values used in persistence analysis are between 0.05 and 1. The concavity tolerance \( \tau \) is 0.05. The average size of the decomposition is 12 (max 18 and min 9).

Figure 15: MPEG 7 beetle. The average size of the beetle polygons is 1690. In all decompositions, the \( \alpha \) values used in persistence analysis are between 0.05 and 1. The concavity tolerance \( \tau \) is 0.05. The average size of the decomposition is 8 (max 11 and min 7).