Combinatorial Shape Decomposition

Ralf Juengling and Melanie Mitchell

Department of Computer Science
P.O. Box 751
Portland State University
Portland, Oregon 97207-0751

Abstract. We formulate decomposition of two-dimensional shapes as a combinatorial optimization problem and present a dynamic programming algorithm that solves it.

1 Introduction

Identifying a shape's components can be essential for object recognition, object completion, and shape matching, among other computer vision tasks [1]. In this paper we present a novel shape-decomposition algorithm, aimed at capturing some of the heuristics used by humans when parsing shapes.

In 1984, Hoffman and Richards [2] proposed the *minima rule*, a simple heuristic for making straight-line cuts that decompose a given shape (or silhouette): Given a silhouette such as the one in Fig. 1(a), the end-points of cuts should be negative minima of curvature of its bounding contour (Fig. 1(b)). Note that this rule does not specify which pairs of these points should be connected to make cuts. Fig. 1(c) gives one possible set of cuts connecting negative minima.

Later, Singh, Syranian, and Hoffman [3] proposed an additional simple heuristic, supported by results of psychophysics experiments on human subjects, called the *short-cut rule*: If there are several competing cuts, select the one with shortest length. For example, in Fig. 1(c), most people would prefer the cuts shown as compared with a cut between the two topmost black dots, which would be significantly longer.

Singh et al. make clear that the minima and short-cut rules are not the only necessary heuristics for shape decomposition; other possible heuristics could involve local symmetries or good continuation, or rely on prior knowledge about the shape's category. However, the two simple heuristics seem to explain many of the experimental results on people.

In this paper we propose an efficient algorithm for shape decomposition that results in these two heuristics being approximately satisfied, without having to compute boundary curvature. Given a polygonal description of a silhouette, our algorithm computes the constrained Delaunay triangulation of the shape, and chooses among the interior edges of this triangulation an optimum set of cuts by solving a corresponding combinatorial optimization problem.

G. Bebis et al. (Eds.): ISVC 2007, Part II, LNCS 4842, pp. 183–192, 2007.

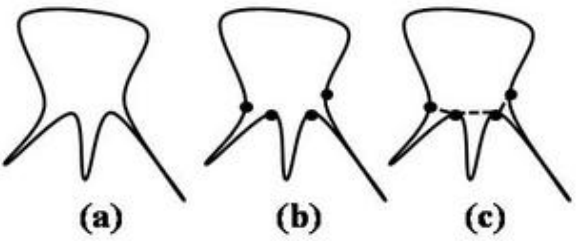

Fig. 1. (a) Silhouette. (b) Black dots mark points at which curvature has a negative minimum. (c) Three possible cuts based on these points.

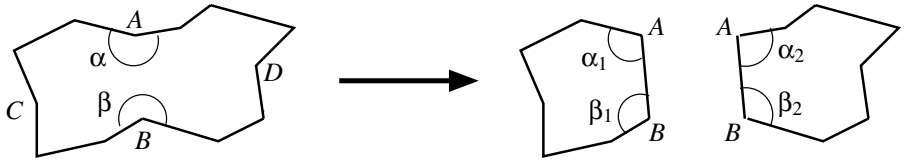

Fig. 2. Making a cut means breaking a polygon into two polygons. Here the cut is made between two concave vertices A and B. Because $\alpha = \alpha_1 + \alpha_2$ and $\alpha < 2\pi$ the number of concave vertices never increases through a cut.

2 Finding Cuts by Combinatorial Optimization

We assume a polygonal description of the shape and require that a cut (1) connects two polygon vertices, and (2) does not cross a polygon edge. It follows that there is only a finite number of possible cuts for any given polygon. Our strategy is to define an objective function over the set of possible cuts, $\mathcal{C}$, and to select the subset of cuts that minimizes the objective function. This is a combinatorial optimization problem with a number of possible solutions exponential in $|\mathcal{C}|$. We introduce a third constraint on the set of possible cuts in Section 2.1 to make the problem amenable to a solution by dynamic programming.

Curvature is not available in our polygonal framework and we need to adapt the minima rule. As in Latecki and Lakaemper [4], vertices with a concave angle play the role of boundary points of negative curvature (we measure the inside angle and call vertices with an angle greater than π *concave*; see Fig. 2).

As Fig. 2 illustrates, placing a cut amounts to breaking a polygon in two. Our objective function favors a decomposition into convex parts by penalizing concave vertices. It is is basically of the form $\sum_k f(\theta_k)$, where θ_k ranges over all angles of a given partition or cut set. For the example in Fig. 2 the difference of the objective function values between the empty cut set (left) and the set $\{AB\}$ (right) is therefore

$$f(\alpha) - f(\alpha_1) - f(\alpha_2) + f(\beta) - f(\beta_1) - f(\beta_2) \tag{1}$$

Thus f should be such that this sum is negative when the cut AB is considered desirable. We will resume discussing the objective function below in Section 2.2.