An Architecture for Region Boundary Extraction in Raster Scan Images Suitable for VLSI Implementation

James M. Apffel,* K. Wayne Current, Jorge L. C. Sanz,t and Anil K. Jain
Department of Electrical Engineering and Computer Science and Computer Vision Research Laboratory, University of California, Davis, California, USA

Abstract. A novel hardware architecture for extracting region boundaries in two raster scan passes through a binary image is presented. The first pass gathers statistics regarding the size of each object contour. This information is used by the hardware to allocate dynamically off-chip memory for storage of boundary codes. In the second raster pass the same architecture constructs lists of grid-joint codes to represent the perimeter pixels of each object. These codes, referred to variously as “crack” codes or “raster-chain” codes in the literature, are later decoded by the hardware to reproduce the ordered sequence of coordinates surrounding each object. This list of coordinates is useful for a variety of shape recognition and manipulation algorithms that utilize boundary information. We present results of software simulations of the VLSI architecture, along with measurements on the coding efficiency of the basic algorithm, and estimates of the overall complexity of a proposed VLSI chip.

Key Words: segmentation, boundary-coding, raster scan, VLSI, architectures

1 Introduction

A variety of approaches to performing image segmentation has been considered, including spatial clustering, region growing, and split-and-merge algorithms (Haralick et al. 1985; Mitiche et al. 1985; Samet et al. 1982). Boundary-based segmentation methods for bitonal images are a subclass in which regions of connected 1’s are separated by their contours. These algorithms typically detect edge transitions in the image and then extract from the contour parameters that are useful for subsequent processing. Since the locations of 1 and 0 transitions in a binary image contain all the information relevant to represent the objects, the extraction of these points is usually the first step in boundary-based segmentation. The transition points can be provided in a line-by-line order, as in run-length encoding techniques, or in a sequential list, as in contour following methods. A sequentially ordered list of the boundary coordinates is especially attractive because it facilitates many kinds of useful processing, such as segmentation, analysis, synthesis, and recognition. For example, individual objects can be manipulated graphically, features such as perimeters, corners, and moments can be extracted, and pattern-matching techniques can be applied (Ellis et al. 1979; Pagliaroni and Jain 1985; Pagliaroni and Jain 1986; Person and Fu 1977; Price 1984).

Historically, there have been two basic approaches to the extraction of boundary points from objects in binary images. The first uses a contour following method analogous to a person traveling the perimeter of a wall. The second uses a technique borrowed from algorithms that label objects in raster-scanned images (Milgram 1979; Sobel 1978). Before describing our chosen algorithm and architecture, we review in the following sections issues that are relevant to the hardware implementation of each method.

1.1 Boundary Following Approaches

A clear advantage of boundary following algorithms is that they are intuitively easy to understand. They also do not require complex data structures or even the storage of boundary codes, as lists of ordered coordinates can be produced directly while a contour is being tracked.

When addressing hardware issues, however, a
number of potential bottlenecks must be resolved. The most serious of these for VLSI implementa-
tions is the need for random access to neighboring pixels. Such access occurs during contour tracking, when the follower may need to visit pixels to the right, left, top, or bottom of the current pixel location. Since the process is “random” or irregular, it is more difficult to implement in fast, pipelined hardware than is the case for raster-scanning methods (Ruetz and Brodersen 1988). A possible solution to this problem is a memory storage structure that permits parallel transfers of two-dimensional blocks of adjacent pixels.

A second difficulty with VLSI contour followers is their requirement to mark boundaries that have already been tracked. A marking is necessary because after it closes a contour, a follower must not trace that contour again. To achieve this marking, one can increase the bit depth of the original image and insert a special tag into locations that have already been visited. Unfortunately, this then creates the need for write-access to a deeper image, which becomes impractical for processing large images at real-time rates. Another approach, suggested in Morrin (1976), would be to peel continuously the boundary of an object until it is erased from the image. Of course, this would be prohibitively expensive in processing time for large objects. Still another approach, used in D’Amato (1982), and Ruetz (Tokyo 1986), would be to save polygonal information about tracked objects. This could be in the form of a convex hull or enclosing rectangle, but if too simple, it could lead to missed contours, especially for cases with nested objects and holes.

1.2 Raster-Scan Methods
An alternative approach to following the boundaries of objects is to collect edge information in a raster-
scan fashion (Cederberg 1979; Chakravarty 1981; Daniellsson 1982; Lindgren 1983; Luhscher and Beddoes 1989; Milgram 1979; Sobel 1978). Algo-
rithms of this type use pixel information from previous scan lines to determine connectivity and thereby label all contours. Raster-scan methods generally require more sophisticated “bookkeeping” than direct contour tracking, but they are attractive for hardware implementation because of inherent regularities in data manipulation. These algorithms also couple well with the current generation of solid-state sensing devices, such as charge-coupled devices, as well as with raster display devices, since these provide and utilize pixels in a raster-scan order.

Raster-scan boundary coding shares most of the hardware advantages of general raster-scan algo-
rithms. Pixel access is regular and sequential, which facilitates straightforward data paths and faster clock speeds. By including the now common line-
delay unit, illustrated in Figure 1, a fixed size, small neighborhood of pixels is examined each clock cycle and an update decision is made. This processing lends itself readily to table look-up methods and pipelined implementations (Milgram 1979; Sobel 1978). The source image is often passed through without a need to add markers to any image, thus overcoming a common drawback of boundary followers.

The most serious disadvantage of raster-scan algorithms is their use of fairly complicated data structures. Since object information is accumulated line by line and for many objects in parallel, a good deal of bookkeeping is necessary to track contours. The need to manipulate and organize these data structures presents formidable challenges in designing a VLSI implementation. Most raster-scan algo-
rithms either do not address these difficulties or pre-
sume a hardware intelligence and organization that is rarely achievable at the chip level.

In this paper we describe a new architecture suitable for VLSI integrated circuit implementation that will produce the edge coordinates of each object in a binary image. The proposed chip set is called the “RASBOC,” an acronym for the raster...