Hardware Design of Shape-Preserving Contour Tracing for Object of Segmented Images

Roy Chaoming Hsu*, Yaw-Yu Lee, Bin-Wen Kao, and Din-Yuen Chan

Department of Computer Science and Information Engineering
National Chiayi University
300 University Road, East District Chiayi City, Chiayi, Taiwan
{rchsu,s0960412,dychan}@mail.ncyu.edu.tw,
goodmorning7337@yahoo.com.tw

Abstract. A hardware design of shape-preserving contour tracing for objects in a segmented image for robot vision and pattern recognition applications is presented in this paper. The proposed contour tracing consists of two processes, namely progressive boundary linking (PBL) and synchronous redundancy pruning (SRP). The proposed method is realized on a System on Chip (SoC) with open-source processor IP core, LEON3, utilizing AMBA bus. Implementation results show that the proposed design achieves better results in terms of accuracy and hardware efficiency comparing with other existing boundary extraction methods such as morphological boundary extraction.

Keywords: Boundary Extraction, Contour Tracing, SOC, AMBA bus.

1 Introduction

Boundary extraction is a technique for accurately extracting the boundary of an object from a digitally-processed image. Such techniques are widely applied for preprocessing purposes in many image/video multimedia applications, including pattern recognition based on shape features, target tracking using object gestures, shape coding, and so forth. Hardware realization of boundary extraction enables extraction of important object features of an image in real-time such that it is of important practice in industrial and medical applications such as vision-based robot and medical imaging. Modern multimedia applications commonly involve the use of image/video segmentation, compression and even simple digitization routines, which inevitably introduce noise and distortion into the original images. As a result, the quality of subsequent image/video automatic recognition and shape-coding applications, in which the object boundary information is of fundamental importance, is inevitably degraded. When applying common contour extraction methods to noisy object boundaries, it is generally found that morphology-based methods provide more accurate results than those obtained using edge-detection schemes. However, visible distortion of the object profile is often apparent when morphology methods based on nonlinear filters (conventionally referred to as structuring elements (SE's)) are applied. Such methods not only suppress

* Corresponding author.

undesirable noise within the image, but also inadvertently eliminate nontrivial details such as textural branches, cracks, shape contour features, and so forth. Therefore, there is a practical need to develop robust contour tracing schemes which effectively suppress the noise content of an image, while simultaneously retaining the fine details of the object contours. In this paper, a hardware design of shape-preserving contour tracing for objects in a segmented image for vision-based robot and pattern recognition applications is presented. The proposed contour tracing consists of two processes, namely progressive boundary linking (PBL) and synchronous redundancy pruning (SRP). The proposed method is realized on FPGA and the simulation result is compared with that of a morphological method.

2 Related Work

FPGA technology [9] for digital design is much more efficient in fast realization of multimedia applications not only in computational cost but also in fast prototyping toward a complete IC. To improve computational efficiency, in [10] a hardware arithmetic ability of ALU is presented utilizing FPGA to realize the complex algorithm and to reduce hardware cost. Hardware realization of noise/redundancy removal is often found in recent literatures of image processing [1, 2]. A FPGA implementation of median filter is presented in [1], where the median filter finds the median value by sorting the image pixels within a 3×3 mask and replaces the value of the center pixel with the median one. By using vertical-horizontal-diagonal sorting [1], hardware space in the realization is hence reduced. In [2], by adding parallelism into the architecture of vertical-horizontal-diagonal sorting, better performance within each clock cycle for the noise removal is achieved. Edge detection is a simple method for extraction of object's boundary of an image. The SUSAN algorithm is developed by parallel modules in order to improve its performance for real-time edge and corner detection process. FPGA architecture of edge and corner detection based on the SUSAN algorithm is presented in [3] to achieve real-time processing of boundary extraction. Mathematical morphology has been an important methodology on image processing [14]. By using basic morphological operations, such as dilation or erosion, and utilizing structuring element as processing unit, noise cleaned and boundary extracted images are produced [5-7]. Morphological cleaning is a process to remove noise of image, by using erosion following by dilation, and then dilation following by erosion to produce a noise-cleaned image. Owing to its simplicity, the morphological operations are often implemented using hardware. In [8], the authors aim at optimizing performances of morphological algorithms with respect to high throughput real-time execution on FPGA. The implemented morphological algorithms were then integrated for complex applications such as vision-based robots or real-time processing and displaying of video flows. It is commonly acknowledged that hardware architecture design is an important factor for improving algorithm complexity. Parallelism of well-pipelined architecture can accelerate the computing time of algorithm and reach real-time requests for most applications [11-13]. In this paper, we focus on the hardware design of a shape-preserving contour tracing [15] for robot vision and pattern recognition applications. The proposed method, hardware

design architecture, and simulation result using a development board are described in the following.

3 Shape-Preserving Contour Tracing

The proposed shape-preserving contour tracing consists of two processes, namely progressive boundary linking (PBL) and synchronous redundancy pruning (SRP) [15].

3.1 Progressive Boundary Linking (PBL)

Process of PBL consists of the following steps.

1. Determining a boundary point of object from either vertical or horizontal direction in a segmented image. The determined boundary point is the initial contour point for starting with the contour tracing.
2. Adding the coordinate of the detected contour point to a contour array to construct the contour image.
3. Moving the mask onto the current detected contour point as the center point, using policy of maximum priority selection, whose function will be described later, to decide the next contour point, and adding the decided next contour point to the contour array.
4. If no new contour point can be decided in Step 3, PBL is finished, otherwise Synchronous Redundancy Pruning (SRP) will be executed, as will be described next, along the contour tracing to determine and remove any redundant points, the contour tracing goes back to step 3 for next iteration to find the next contour point.

3.2 Synchronous Redundancy Pruning (SRP)

Whenever a point is determined as a contour point, SRP will start to execute the following two phases to improve the definition of the contour.

1. Whether previous points have crack in between or not is checked in the first phase. Herein, the previous points are defined as the previous three points of the new contour points. If there is crack, the crack is filled with the same bi-level value of the nearby contour point and the contour array is updated. If no cracks exist between previous points, SRP continues in executing phase 2.
2. If SRP determines there is branch formed by previous points, SRP will remove the redundancy by pruning the branch and updates the contour array. Hence a redundancy-free contour path can be constructed by the SRP process.

The contour tracing then goes back to PBL to find next contour point, and redundancy, if exists, will be checked and removed by SRP to finally obtain a shape-preserving contour image.

4 Hardware Architecture

Fig. 1. shows the proposed hardware design of the shape-preserving contour tracing. The architecture is composed of main controller, and PBL and SRP modules.

Fig. 1. Hardware architecture of the proposed method

4.1 Main Controller

Main controller's function is to control timing of access to memory and loading and storing of data input and output for processing. The main controller reads input image pixels as an array of $M \times N$ in the main SRAM memory and initializes the contour tracing by utilizing a finite state machine (FSM) as shown in Fig. 2.

Fig. 2. Finite state machine of the main controller

In state S0 to S8, image pixels in a 3×3 mask is accessed sequentially and the process modules will start to process the contour tracing at state S9. After the contour point is found, the main controller stores the result to a contour array of the SRAM at state S10 and resumes contour tracing for next boundary point. The main controller will continue tracing contour until a closed contour is constructed.

4.2 PBL Module

PBL module is composed of two sub-modules, priority calculation and decision module and selection of maximum priority point module. The PBL first calculates the surrounding points priorities of the central boundary point by priority calculation and decision module, then the priorities between the surrounding points are compared to decide which point is selected as the next contour point by selection of maximum priority point module.

Priority calculation and decision module

This module calculates priority of 8 candidate contour pixels surrounding the current contour points within the 3×3 mask. The rule of priority calculation and decision consists of three steps. First, if any one of 8 neighboring point of the current boundary

point is 1, it will be considered as a candidate point of the next contour point. Second, the two neighboring points of the candidate point are checked whether they have only one pixel equals to 0 [15]. If it is, the output value is set as 0 (first priority). If the value of both the neighboring pixels equal to 0, the output is set as 1 (second priority), otherwise the candidate point is considered as “don’t care” and will not be candidate anymore. To better explain how the priority calculation module functions, the relationship between candidate point and its two neighboring points and how the output is decided are shown in Fig. 3. In Fig. 3(a), only one of the two neighboring points of the underlined point is 0, and the output will be set to 0, while in Fig. 3(b) and 3(c) the output is set to 1 and “don’t care”, respectively. Third, the output value of the candidate points are added with it’s occurrence times (OT) value, the number of times the candidate is decided as a boundary point and is stored in memory previously, using an adder to obtain the final priority vale as below

$$Priority = OT + output \tag{1}$$

Fig. 3. Second step of the rule of priority calculation. (a) Output of Underlined point is 0, (b) Output of Underlined point is 1, (c) Output of Underlined point is “don’t care”.

In this study, the value of OT is 0 and 1, with lower value represents larger priority, which represents for not being decided as boundary and already being decided as boundary point once, respectively. The combination of priority output with OT will constitute with two bits in (OT,output) form, (OT,output)= (0,0) means output priority is 0 and the candidate point is not yet decided as boundary such that this candidate point has the maximum priority for next boundary point. Fig. 4. show the hardware design of the priority calculation module, where In1 and In2 represent the value of two neighboring points of the candidate point and there are eight modules constructed in parallel for the 8 neighboring points of the current boundary point. The enable bit is from the candidate boundary point.

Selection of maximum priority module

This module will select the point with maximum priority as output. The priority comparison is done by using 2-input comparator sequentially, and the point with the maximum priority is selected as next contour point and stored in the contour array.

4.3 SRP Module

Whenever PBL determines a new contour point, SRP will start to check whether the previous contour points are cracks or branches and the cracks and branches are subsequently be filled or removed, respectively, to improve the definition of contour

Fig. 4. Hardware design of single priority calculation and decision module

along the contour tracing. Fig. 5 shows how SRP module works on redundancy pruning by crack filling and branch removing, where P_i is the current contour point, $P_{i-1}, P_{i-2}, P_{i-3}$ are previous contour points in order before P_i , with each point accessed from the contour array in the memory. Fig. 5(a) is the original image where distance between P_i and P_{i-3} is 2. In Fig. 5(b) SRP executes Phase 1, the crack is filled by replacing 1 with 0. Distance between P_i and P_{i-3} is 1 now and a branch exists. Fig. 5(c) shows that SRP executing Phase 2 to remove branch by replacing 1's, i.e., P_{i-1}, P_{i-2} , with 0's. In executing the SRP module, the distance of P_i and P_{i-3} , needs to be realized with the distance are defined as below.

Fig. 5. SRP Execution examples (a) Original image with Distance (P_i, P_{i-3})=2, a crack exists. (b) the result after executing SRP Phase 1, Distance(P_i, P_{i-3})=1 (c) SRP executes Phase 2 by removing the branches to obtain the final result.

$$Distance(P_i, P_{i-3}) = \max(|x_i - x_{i-3}|, |y_i - y_{i-3}|) \quad (2)$$

If the distance equals to 2, SRP decides that crack exists and execute phase 1 to fill the crack by storing a 1 to the positions between P_i and P_{i-3} of the contour array. If there is no crack, SRP goes to phase 2. In Phase 2, if the distance between P_i and P_{i-3} equals to 0 or 1, branch exists and the branch is then removed and SRP ends. Fig. 6 shows the hardware design of SRP module. In executing SRP, the redundancy is pruned by filling the crack or removing the branch and the contour array in the memory is updated. The contour tracing goes back to PBL and SRP for next contour point.

Fig. 6. Hardware design of SRP module

5 Simulation Results

The proposed FPGA design for contour tracing is realized on a System on Chip (SoC) with open-source processor IP core, LEON3, and AMBA bus, using Verilog hardware description language and the performances are compared with an existing morphological noise cleaning and contour extraction method. The LEON3 is a synthesizable VHDL model of a 32-bit processor compliant with the SPARC V8 architecture, which is highly configurable, and particularly suitable for system-on-a-chip (SOC) designs. GR-XC3S-1500 development board, from Gaisler Research and Pender Electronic Design [17], is used in realization of the contour tracing. The board incorporates a Xilinx Spartan3-1500 FPGA, and is capable of operating stand-alone from a single +5V power supply.

The existing morphological method is composed of morphological cleaning and boundary extraction. The morphological cleaning is performed by a closing following an opening using a cross-shaped SE, while the morphological boundary extraction is performed by taking the difference between the eroded and original noise-cleaned image. In realization of the morphological method, a low complexity architecture for binary image erosion and dilation is used [16] by decomposing a 3×3 symmetric structuring element into structuring elements of 1×3 (SE1) and 3×1 (SE2). The equation for erosion and dilation is shown in (3) and (4), respectively.

$$Erosion(A, SE) = (Erosion((Erosion(A, SE_1), SE_2)) \quad (3)$$

$$Dilation(A, SE) = \sim (Erosion((Erosion(\sim A, SE_1), SE_2)) \quad (4)$$

In low complexity architecture, only erosion hardware is required to be designed, while the dilation's structure will be designed by utilizing the erosion hardware with two additional NOT gate (inverse) to reduce the hardware cost per (4). Fig. 7 shows the 1-D erosion of the low complexity architecture by the 1×3 structuring element.

Fig. 7. Hardware structure of 1-D erosion

In Fig. 7, the variable Sum is initialized with 0, and the image pixel is fed in sequentially and is decided whether it is 0 or not. If input is zero, Sum will be reset to 0, otherwise it will be incremented by the adder and the result will be compared with 3. The output will be 1 if the sum equals to 3, otherwise the output will be zero. The 1-D erosion can be realized using two XORs, two ANDs, and two multiplexers. A 2-D erosion or dilation is then realized by utilizing by a 1-D erosion along the x-axis following a 1-D y-axis erosion. The morphological contour extraction is produced by the XOR gate of noise-cleaned image and the eroded morphological noise-cleaned image. The contour extraction equation is the following

$$EdgeExtract(A) = XOR(A, erosion(A)) \quad (5)$$

where A is the morphological noise-cleaned image. The performances of the proposed shape-preserving contour tracing and mathematical morphology method are evaluated using three 352×288 test images shown in Fig. 8(a), namely “Dog”, “Hammer” and “Vase”. To demonstrate the capabilities of the proposed contour tracing in shape preservation and redundancy suppression, random noises with bit error rate of 1% are inserted in the test images as shown in Fig. 8(b). Fig. 8(c) and Fig. 8(d) show the experiment result obtained by the morphological method and the shape-preserving contour tracing, respectively.

	Original Image	Orig. Image with 1% random noise	Morphological Method	The Proposed Method
Dog
Hammer
Vase
	(a)	(b)	(c)	(d)

Fig. 8. (a) Original image. (b) Noisy image by adding random noise with bit error rate of 1% to the original image. (c) Experiment result obtained by the morphological method. (d) Experiment result obtained by the shape-preserving contour tracing.

In the performance evaluation, three evaluation metrics are adopted for comparing experimental results of the proposed method and morphological method, which are accuracy, total execution time, and memory requirement. *Accuracy measurement:*

The baseline-based mean absolute difference (BMAD) [15], as given below, is used for accuracy measurement,

$$BMAD = \frac{1}{N_p} \sum_{j=1}^{N_p} \left| \hat{d}_j - \bar{d}_j \right| \text{ for } \hat{d}_j \neq \infty \wedge \bar{d}_j \neq \infty \quad (6)$$

where $\hat{d}_j$ and $\bar{d}_j$ are the orthogonal distances measured from the j^{th} sampling point of the reference baseline axis to the traced contour and the accurate contour, respectively. According to Eq. (6), the overall quality of the extracted contour is

evaluated by computing the mean of individual baseline-based absolute difference obtained for each of the N_p measurements. When the *BMAD* is lower, it means that the traced contour and the accurate contour will have lower mean distance error, and the accuracy between the traced and accurate is relatively higher.

Without lose of generality, the accuracy is defined as $(1 - BMAD) \times 100\%$. Table 1 shows the accuracy of the results obtained by morphological method and the proposed method. From inspection, the accuracy for Fig.8(d) is saliently larger than that obtained for Fig. 8(c), which exhibits that the proposed method outperforms the morphological boundary extraction following morphological cleaning process in precisely delineating object’s contour under the environment with random noises.

Table 1. Accuracy comparison between morphological method and the proposed method

	Morphological Method	The Proposed Method
Dog	25%	98%
Hammer	30%	99%
Vase	33%	99%

In the morphological method, the boundary extracted image is given by taking the difference image between the eroded image and the original morphological cleaned image [14], and consequently, this technique tends to “shrunk” the object’s shape and boundary when a 3×3 square SE is used. As a result, the extracted boundary pixels are “eroded” with respect to the actual contour pixels. Moreover, the morphological cleaning operations lead to noticeable distortions on semantic object’s significant details and concavities such as that occurring at “Dog” ears areas in Fig. 9(a), “Hammer” neck in Fig. 9(b), and “Vase” top in Fig. 9(c).

Fig. 9. Magnified portions of boundary extraction results. (a) Dog’s ear, (b) Hammer’s neck, (c) Vase top.

Total Execution Time Measurement

Assuming that P boundary points are extracted in our proposed method, which means that PBL and SRP executes P times. According to PBL and SRP, it takes 7 and 1 clock cycles, respectively, for PBL and SRP to decide a boundary point such that the total execution of the proposed method is $8P$, which is proportional to the number of P as below.

$$T_{proposed} = 8P \propto P \quad (7)$$

In realization of the mathematical morphological method using low complexity architecture, the total execution time can be reduced which is proportional to $M \times N$, size of the image, for a single erosion operation as below

$$T_{low-complexity} \propto M \times N \quad (8)$$

It requires two erosions and two dilations in realization for a single morphological cleaning and one erosions and one XOR operations in realization for morphological boundary extraction such that the total execution time of the low complexity morphological method is about $6 \times M \times N$, which is proportional to the size of the image. Table 2 shows the total execution time measured from the main controller using a software timer. From inspection, it is apparent that our proposed method obtains fast execution result for boundary extraction (i.e. lower total execution time) than the morphological method in every case. Based upon the evidence provided in Table 2, it seems reasonable to verify that the longer execution time of the morphological method is the result primarily of processing the whole image.

Memory Requirement

Memory utilization, such as logic and storage utilization, is commonly acknowledged metric in measuring the hardware realization of an algorithm. The flip-flop and latch are the basic storage unit, while 4-input look-up table (LUT) is the logic component for calculation. Besides, gate count is also a good measurement for space cost of a hardware realization. Hence, flip-flop, latch, 4-input LUT, and gate count are used in comparing the realization of the proposed method and the morphological method. Table 3 shows the metrics generated by running ISE Foundation, a FPGA design tool from Xilinx. From inspection, the 4 measuring metrics of the proposed method are in comparable of the morphological method, which means the hardware cost of the two methods are almost the same. This comparison exhibits that under the same hardware cost the proposed method outperforms the morphological method in accuracy and execution time. Significantly, it is precisely the fast execution time and accuracy of boundary extraction which are of most importance in realizing the contour extraction of the automatic robot vision and pattern-recognition applications used to analyze medical images, remote sensing images, and so forth. Hence, the present results suggest that the proposed method is in practical use as a preprocessing technique for such applications.

Table 2. Total execution time measurement of morphological method and the proposed method

	Morphological Method	The Proposed Method		
Total Execution time (in 0.1 ns)	4,159,301	Dog	Hammer	Vase
		32,480	33,768	29,841

Table 3. Memory requirement comparison of morphological method the proposed method

Number of	Morphological Method		The Proposed Method	
Slice registers	Flip-flips	Latches	Flip-flips	Latches
	4,382	4	4,409	4
4-input LUTs	14,143		14,177	
Gate count	1,458,474		1,458,879	

6 Conclusions

In this paper, a hardware design of shape-preserving contour tracing for object in a segmented image for robot vision and pattern recognition applications is presented. The proposed contour tracing consists of progressive boundary linking and synchronous redundancy pruning processes. Simulation results by realizing the proposed method on SoC shows that the proposed design achieves better results in terms of accuracy and hardware efficiency comparing with morphological boundary extraction. The present results suggest that the proposed method is in practical use as a preprocessing technique for automatic robot vision and pattern-recognition applications used to analyze medical images, remote sensing images, and so forth.

References

1. Bates, G.L., Nooshabadi, S.: FPGA Implementation of a Median Filter. In: Proc. of IEEE on Speech and Image Technologies for Computing and Telecommunications, pp. 437–440 (1997)

2. Hazra, A., Bhattacharyya, J., Banerjee, S.: Real Time Noise Cleaning of Ultrasound Images. In: Proc. of IEEE Symposium on Computer-Based Medical Systems, pp. 379–384 (2004)

3. Torres-Huitzil, C., Arias-Estrada, M.: An FPGA Architecture for High Speed Edge and Corner Detection. In: Proc. of IEEE International Workshop on Computer Architectures for Machine Perception, pp. 112–116 (2000)

4. Hussmann, S., Ho, T.H.: A High-speed Subpixel Edge Detector Implementation Inside a FPGA. Real-Time Imaging 9(5), 361–368 (2003)

5. Jun, Y., Li, X.: Boundary Detection Using Mathematical Morphology. Pattern Recognition Letters 16(12), 1277–1286 (1995)

6. Shih Frank, Y., Shouxian, C.: Adaptive Mathematical Morphology for Edge Linking. Information Sciences 167(1), 9–21 (2004)

7. Dong, P.: Implementation of Mathematical Morphological Operations for Spatial Data Processing. *Computers & Geosciences* 23(1), 103–107 (1997)
8. Baumann, D., Tinembart, J.: Designing Mathematical Morphology Algorithms on FPGAs: An Application to Image Processing. In: Gagalowicz, A., Philips, W. (eds.) CAIP 2005. LNCS, vol. 3691, pp. 562–569. Springer, Heidelberg (2005)
9. Ali, K.S.: Digital Circuit Design using FPGAs. *Computers & Industrial Engineering* 31(1), 127–129 (1996)
10. Gustin, V.: An FPGA Extension to ALU Functions. *Microprocessors and Microsystem* 22(9), 501–508 (1999)
11. Iakovidis, D.K., Maroulis, D.E., Bariamis, D.G.: FPGA Architecture for Fast Parallel Computation of Co-occurrence Matrices. *Microprocessors and Microsystems* 31(2), 160–165 (2007)
12. Battle, J., Marti, J., Ridao, P., Amat, J.: A New FPGA/DSP-Based Parallel Architecture for Real-Time Image Processing. *Real-Time Imaging* 8(5), 345–356 (2002)
13. Oswaldo, C., Graham, M.: A Clocking Technique for FPGA Pipelined Designs. *Journal of Systems Architecture* 50(11), 687–696 (2004)
14. Gonzalez, R.C., Woods, R.E.: *Digital Image Processing 2/e*. Prentice-Hall. Inc., New Jersey (2002)
15. Chan, D.-Y., Hsu, R.C.: Robust Shape-Preserving Contour Tracing with Synchronous Redundancy Pruning. *Pattern Recognition Letters* (29), 569–579 (2008)
16. Hedberg, H., Kristensen, F., Nilsson, P., Owall, V.: A low complexity architecture for binary image erosion and dilation using structuring element decomposition. In: *IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 3431–3434 (2005)
17. Gaisler Research, <http://www.gaisler.com>