LETTER *Special Section on Parallel and Distributed Computing and Networking*

An Efficient $O(1)$ Contrast Enhancement Algorithm Using Parallel Column Histograms

Yan-Tsung PENG[†], ***Student Member***, **Fan-Chieh CHENG**[†], ***and*** **Shanq-Jang RUAN**[†a)], ***Members***

SUMMARY Display devices play image files, of which contrast enhancement methods are usually employed to bring out visual details to achieve better visual quality. However, applied to high resolution images, the contrast enhancement method entails high computation costs mostly due to histogram computations. Therefore, this letter proposes a parallel histogram calculation algorithm using the column histograms and difference histograms to reduce histogram computations. Experimental results show that the proposed algorithm is effective for histogram-based image contrast enhancement.

key words: *contrast enhancement, parallel histogram construction, constant time*

1. Introduction

It is pervasive and of an essential part in our lives nowadays that we uses high-resolution display devices. Hence, contrast enhancement plays an important role to further improve visual quality in those devices. Based on two-dimension (2-D) histogram calculation, a contextual and variational contrast enhancement (CVC) algorithm was presented to improve image quality [1]. However, the calculation of 2-D histogram definitely entails high time complexity.

To solve this problem, numerous methods can be employed to calculate histogram [2]–[4]. One of the prior works used the past kernel histogram with modifying the boundary information [2] for acceleration, but it still involved $O(r)$ time complexity, where r is the kernel radius. Inspired from integral images, a constant time $O(1)$ method uses a superset of the cumulative image formulation, named integral histogram, to compute the kernel histogram [3], while it involves high space complexity. In order to avoid utilizing huge storage, the distributive method computes column histogram that disjoint column regions for kernel histogram [4]. According to its characteristics, the kernel histogram can be easily computed. In this letter, the computation cost of column histograms can be further reduced using the proposed difference histograms.

2. Modified CVC Algorithm

Fig. 1 snapshots the principle of the proposed method for column histograms. The kernel histogram contains the visited pixel and its neighbours. It can be disjointed to $(2r+1)$ column histograms shown in Fig. 1 (a). As the visited pixel is changed, the kernel histogram is easily updated by adding the rightmost column histogram and subtracting the leftmost column histogram shown in Fig. 1 (b). In order to further reduce the computation cost, the difference information can be stored in difference histogram, which is expressed as follows:

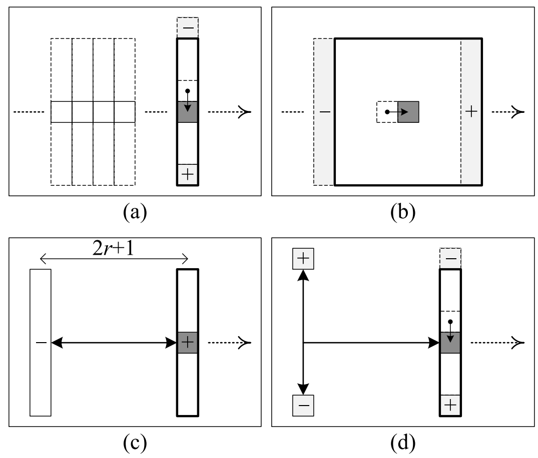

Fig. 1 Histogram construction using column histograms: (a) traditional column histograms; (b) updating kernel; (c) new column histograms; (d) updating new column histograms.

$$D_x = C_{x+r} - C_{x-r-1}, \tag{1}$$

where D is the difference histogram and C is the column histogram.

In Fig. 1 (c), one past column histogram must be subtracted from the visited one. After initialization, each pixel does not contain the traditional column histogram but the differential column histogram. Compared to the traditional one, the proposed column histogram further adds left-top pixel and subtracts left-bottom pixel visualized in Fig. 1 (d). However, its computation cost is much lower than the subtraction operator in Fig. 1 (b). Suppose that $I(x, y)$ is the input image, and K is kernel histogram per pixel. The pseudocode is then listed below:

for $y \leftarrow 1$ to h **do**
  for $x \leftarrow 1$ to w **do**
    Add $I(x+r, y+r)$ to D_x
    Subtract $I(x+r, y-r-1)$ from D_x
    Add $I(x-r-1, y-r-1)$ to D_x

Manuscript received January 5, 2013.
Manuscript revised April 7, 2013.
[†]The authors are with the Department of Electronic and Computer Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan.
a) E-mail: sjruan@mail.ntust.edu.tw
DOI: 10.1587/transinf.E96.D.2724

Subtract $I(x - r - 1, y + r)$ from D_x
Add D_x to K
end for
end for

Note that D_x is able to be boosted by parallel computation since all column histograms are data independence.

Based on the improved histogram construction method, the 2D target histogram [1] can be constructed in constant time. Let l be the gray-level of each pixel in the image I, the 2D histogram of each input image is formed as

$$\mathbf{T} = \{t(l, k) | 0 \leq l, k < L\}, \quad (2)$$

where L is the number of the gray-level, k is the gray-level occurs in neighborhoods of each arbitrary pixel, and $t(l, k)$ is each element of $\mathbf{T}$. Once the kernel histogram K has been computed using the proposed column histograms, it can be employed to accumulate $\mathbf{T}$. For each visited pixel that has l-th gray level, K is added to $t(l, k)$ in $O(1)$ complexity, where k just represents each element in K.

3. Experimental Results

In general, updating each traditional column histogram needs 1 addition and 1 subtraction, while the kernel computation using traditional column histograms needs 256 additions and 256 subtractions. In contrast, updating the proposed column histogram needs 2 additions and 2 subtractions, but the computation of updating kernel is reduced to only 256 additions.

In order to establish the improvement of the computation in practice, the proposed method and the compared method are implemented in C. All experiments were conducted on the machine with 12GB RAM and two multi-core processors. The processor information includes Intel® Xeon® CPU E5520 @ 2.27GHz with 4 core counts, 8 thread counts, and 8MB cache.

Figure 2 shows the line chart of the execution time for histogram construction. In this coordinate system, x-axis is the radius and y-axis is the execution time (ms). As a result, the proposed method almost reduces half of the computation cost of the compared method [4]. Figure 3 compares the CVC algorithm [1] and the improved one using the proposed method. It is easily observed that the CVC algorithm [1] exhaustively visits all neighbours of each pixel to generate 2-D histogram. Hence, its time complexity is much higher than the use of column histograms. Therefore, the overall performance can be dramatically increased as the CVC algorithm [1] is integrated with the proposed method.

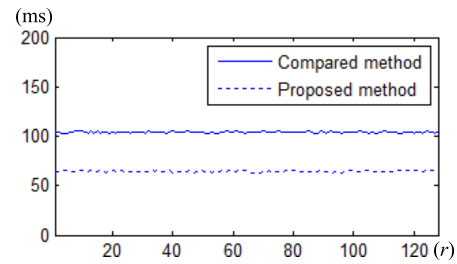

Fig. 2 The execution time to histogram construction using 8 multi-threads.

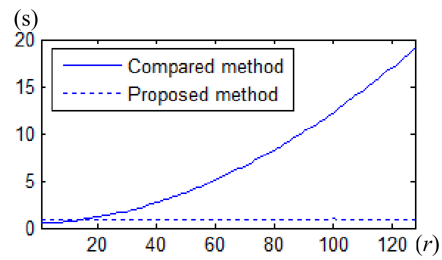

Fig. 3 The execution time to CVC algorithm using different histogram construction methods.

4. Conclusion

This letter has proposed a histogram construction algorithm which obtains $O(1)$ complexity as the window radius varies. Moreover, a parallel implementation has also been shown to verify the high performance of the proposed algorithm with the extent of our knowledge. Our experiments confirmed that the proposed method really decreased the amount of the computation related to histogram-based image processing applications.

Acknowledgements

This work was supported by the National Science Council of ROC under Contract NSC101-2221-E-011-164-MY3.

References

[1] T. Celik and T. Tjahjadi, "Contextual and variational contrast enhancement," IEEE Trans. Image Process., vol.20, no.12, pp.3431–3441, Dec. 2011.
[2] T.S. Huang, G.J. Yang, and G.Y. Tang, "A fast two-dimensional median filtering algorithm," IEEE Trans. Acoust. Speech Signal Process., vol.ASSP-27, no.1, pp.13–18, Feb. 1979.
[3] F. Porikli, "Integral histogram: A fast way to extract histograms in Cartesian spaces," Proc. IEEE 2005 Int. Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.829–837.
[4] S. Perreault and P. Hebert, "Median filtering in constant time," IEEE Trans. Image Process., vol.16, no.9, pp.2389–2394, Sept. 2007.