Microprocessors and Microsystems 39 (2015) 135–143

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

A low latency minimum distance searching unit of the SOM based hardware quantizer

W. Kurdthongmee [*]

Division of Computer Engineering, School of Engineering and Resources Management, Walailak University 222 Thaibury, Tha-sa-la, Nakorn-si-thammarat 80160, Thailand

ARTICLE INFO

Article history:
Available online 16 February 2015

Keywords:
Image compression
Image quantization
Self Organizing map
Latency
Minimum distance searching
FPGA

ABSTRACT

Parts of a SOM (Self-Organizing Map) based quantizer can be performed in parallel; i.e. distance calculation between an input pixel and a group of codewords or processing elements (PEs), and updating weight of PEs. To search for the best matching unit (BMU) whose distance is the minimum, all distances are inevitably required to compare with each other. Conventionally, the minimum distance searching unit is constructed from a group of comparators which are connected in a multistage manner in order to come up with the final single minimum distance and its index. In this way, the overall latency of the unit is linearly proportional to the number of stage of comparators $log_2(C)$ where C is the number of distances. In this paper, we propose a novel hardware centric algorithm with the objective to reduce the latency for the minimum distance searching unit. In a simple form, the algorithm relies on using a memory of K addresses of 1-bit word size where K is equal to the maximum value of distance. During operation, all distances are used to refer to the memory addresses in order to change their states from 'unoccupied' to 'occupied'. To efficiently search for the first address whose state is 'occupied', which is equivalent to the minimum distance, the look up table is employed. The algorithm is also adapted to make it more feasible to realize on an FPGA platform. The synthesis results compared with the conventional minimum distance searching indicate that the FPGA resource requirements of the algorithm are twice in terms of slices and LUT usages. In term of latency reduction, the implementation takes only 0.62 times of the conventional one for a PE size of 256. After integrating the unit to the SOM based quantizer, it has found that the obtained frame rate is 1.50 times of the conventional one for a PE size of 256, the image size of 512×512 and the clock speed of 66.67 MHz. The latency reduction can be further improved if the FPGA supports combining all the 'occupied' states in a single stage in contrast to use a group of internal limited input size LUTs.

1. Introduction

A true color digital image requires 24 bits to specify the color of each pixel on the screen. It is expensive to have a high-speed memory to support such a full-color display on a high resolution display [1,2]. An alternative solution is to provide a limited number of bits for specifying the color of each pixel. Each of these values is then used as an index into a user defined color palette table. The process of carrying out a color palette table selection is known as the "color quantization process." The color quantization is one of the most useful lossy compression methods with the benefit of easy implementation on the image receiver site [3]. Typically, the color quantization attempts to find an acceptable set of palette colors that can be used to represent the original colors of a digital image. There are two points that make this lossy image compression successful. First of all, it exploits the limited ability of human perception which is capable of distinguishing less than a thousand colors. In addition, it exploits the limited capability of many display devices to display true colors on many display devices like a billboard [4] or a low quality LCD (Liquid Crystal Display) which have constantly been used as replacements for text centric display media. Applying the color quantization to an RGB color image frame of size 640×480 pixels, the resulting image with a color palette size of 256 will have only $(640 \times 480) + (256 \times 3) = 307,968$ bytes which is about one-third of the original image size. This makes the color quantization process widely utilized in many applications especially in computer graphics and image processing.

In general, Self Organizing Map (SOM) is one of the most prominent artificial neural network models adhering to the unsupervised learning paradigm [5]. It has been employed to solve problems in a wide variety of application domains. For the applications in the

[*] Tel.: +66 075672318; fax: +66 075672399.
E-mail address: kwattana@wu.ac.th

http://dx.doi.org/10.1016/j.micpro.2015.01.009
0141-9331/© 2015 Elsevier B.V. All rights reserved.

engineering domain, it was elaborately surveyed and reported in [6]. SOM has long been employed to speed up the color image quantization especially in order to implement the quantizer in hardware. In color quantization, the SOM model consists of a number of neural processing elements (PE) or 'codebook'. Each of the PE, i or a 'codeword' i, is assigned a three dimensional weight vector m_i. During the learning or the color palette generation stage, the iteration t starts with the random selection of an input pixel $p(t)$. $p(t)$ is presented to SOM and each PE determines its activation by means of the distance between $p(t)$ and its own weight vector. The PE with the minimum distance is referred to as the winner, m_c, or the best matching unit (BMU) at the learning iteration t, i.e.:

$$m_c(t) = min_i \|p(t) - m_i(t)\|. \tag{1}$$

The $l1$-norm and $l2$-norm are the popular ways to measure the distance between $p(t)$ and a PE's weight vector $m_i(t)$. They are defined by the following equations:

$$d(p(t), m_i(t)) = \sum_{k\in\{R,G,B\}} |p(t)_k - m_i(t)_k|, \tag{2}$$

$$d(p(t), m_i(t)) = \sqrt{\sum_{k\in\{R,G,B\}} (p(t)_k - m_i(t)_k)^2}. \tag{3}$$

Finally, the weight vector of the BMU as well as the weight vectors of selected PEs in the vicinity of the BMU are adapted. This adaptation is implemented as a gradual reduction of the component-wise difference between the input pixel and weight vector of the PE, i.e.:

$$m_i(t+1) = m_i(t) + \alpha(t) \cdot h_{ci}(t) \cdot [p(t) - m_i(t)]. \tag{4}$$

Geometrically speaking, the weight vector of PEs of the adapted units are moved a bit towards the input pixel. The amount of weight vector movement is guided by a learning rate, α, decreasing with time. The number of PEs that are affected by this adaptation is determined by a neighborhood function, h_{ci}, which also decreases with time. This movement makes the distance between these PEs decrease and, thus, the weight vector of the PEs become more similar to the input pixel. The respective PE is more likely to be the BMU at future presentations of this input pixel. The consequence of adapting not only the winner alone but also a number of PEs in the neighborhood of the BMU leads to a spatial clustering of similar input patterns in neighboring parts of the SOM. Thus, similarities between input data that are presented in the n-dimensional input space are mirrored within the two-dimensional output space of SOM or SOM map.

The pixel mapping stage is very similar to the color palette generate stage with some exceptions. That is to say there is no need to perform adaptation to the BMU and its neighbors of the SOM map with respect to the input pixel. Instead, the index of the BMU corresponding to the input pixel is returned and used to encode the input pixel.

It can be said that a SOM quantizer is a good candidate for hardware implementation. This arises from the fact that its PEs can be designed to process the required operations on their own data structures; i.e. a weight vector or a codeword. That is to say each PE can be implemented with its inherent behavior to completely perform either Eq. 2 or Eq. 3 and 4 on its own weight vector m_i. All PEs can be instantiated in such a way to process these operations in parallel. This results in a significant acceleration of the overall execution of the quantizer. In between the operations given by either Eq. 2 or Eq. 3 and 4, it, however, requires a process described by Eq. 1 that involves all PEs to participate. Suppose all PEs, C, can be executed in parallel, the direct hardware implementation of Eq. 1 requires $\sum_{k=0}^{n-1} 2^k$ comparators where $n = log_2 C$. If $C = 256$, the latency of the minimum distance searching for a single pixel (both in color palette generation and pixel mapping stages) is equal to 8 times of the latency within a single comparator at best. For a quantizer with j PEs, the schematic representation for part of the sub-unit responsible for the BMU searching is shown in Fig. 1.

In summary, it can be said that reduction of the number of comparator stage will be benefit for the overall performance of the quantizer. The rest of this paper is organized as follows. In Section 2, we present related work in the area of SOM based quantizer focused on the hardware implementation. This is then followed by the proposed hardware friendly approach to accelerate the minimum distance searching stage of the algorithm in Section 3. In Section 4, the implementation is detailed, the experimental results and discussions are presented. Finally, the paper is concluded in Section 5.

2. Related work

With respect to the hardware implementation of a SOM quantizer, a mixed analog and digital approach was presented in [7]. Its drawback was that the analog techniques tended to increase circuit density and suffered from a lack of accuracy due to the impact of noise. Another hardware oriented implementation of SOM was also proposed by [8] which is targeted to implement on a SoC (System-On-Chip) hardware platform with software/hardware co-design methodology. To rectify the weaknesses of the predecessor hardware-based SOMs, [2] proposed a fully hardware target relying on a digital design for a 3D SOM. The design was based on a single instruction multiple data (SIMD) stream architecture. The proposed technique was implemented in order to evaluate its performance for different sizes of the network on an ASIC (Application Specific Integrated Circuit). The implementation results indicated that 30 frames per second of images of size 512 × 512 with 64 color palette size were obtainable. In [1], a similar architecture was ported to synthesize on an FPGA. Unfortunately, it required 2 Xilinxs Virtex Family of FPGA chips to synthesize the system whose color palette size was limited to only 8! It was reported in [9] the success of a single moderate density FPGA implementation of a SOM with a real-time performance and acceptable image quality. The proposed architecture was based wholly on unsigned integer arithmetic with an operator sharing concept. The maximum obtainable color palette size was 128 × 2 or 256 colors when synthesised on a single XC2VP100 FPGA device running at a maximum frequency of 24 MHz. This was equivalent to a maximum frame rate of 25 frame per second (fps) for an input image of size 640 × 480 pixels. This was attained when image pixels are subsampled during the color palette generation. As an extension to the architecture in [9] which relied on using a winner-take-all (WTA) scheme, [10] proposed a more hardware centric SOM algorithm taking into account a topological relationship among PEs. This improved the convergence rate of the SOM algorithm and, in turn, increased the quality of the quantized image significantly. Apart from only updating the BMU, the newly proposed algorithm updates its surrounding PEs with variable learning rates depending on their relative distances to the BMU. The "update-and-use" scheme was proposed and used in order to guarantee that all PEs were visited only once in the weight update stage. In [11], a novel tri-state rule was used in updating the network weights during the training stage. The rule implementation was claimed to suit the FPGA architecture, and it allowed extremely rapid training. The minimum distance searching part of the implementation also relied on employing the multistage comparators/multiplexers. It was reported that the part required 7 clock cycles. The number of bits to represent the weight vectors was studied by [13]. It was concluded that even for the low resolutions