Hardware Design of Histograms of Oriented Gradients Based on Local Binary Pattern and Binarization

Shen-Fu Hsiao, Jun-Mao Chan, and Ching-Hui Wang
Department of Computer Science and Engineering
National Sun Yat-Sen University
Kaohsiung, Taiwan

***Abstract*—Histograms of oriented gradients (HOG) are widely used to extract image features for pedestrian detection, but the involved complex computations are not amenable for hardware implementation. This paper presents a simplification of HOG using local binary patterns for gradient and magnitude computation. Furthermore, the complex normalization is replaced by simple binarization, leading to significant saving of area cost without sacrificing too much detection rate.**

Keywords—HOG; feature extraction; local binary pattern; pedestrian detection; computer vision

I. INTRODUCTION

Pedestrian detection is an important issue in advanced driver assistance systems (ADAS). Histograms of oriented gradients (HOG) has been widely used to extract features for pedestrian detection [1-11]. However, the HOG involves complex arithmetic computations which are not amenable for hardware implementations. Recently, some papers presents various approaches to reduce the hardware cost of HOG implementation [2-8]. For example, SRA (Square Root Approximation) is adopted to simplify the HOG magnitude calculation and tangent function in orientation calculation is also approximated by sum of power of two in [8]. In [9-11], combination of local binary pattern (LBP) and HOG is proposed to improve the detection rate, but without hardware implementation.

This paper presents a hardware design for HOG based on local binary pattern of gradients and binarization (instead of normalization) to simplify the implementation. Experimental results show that the proposed design can significantly reduce the area cost with only slight degradation in pedestrian detection rate.

II. HOG ALGORITHM

An input image is divided into units of blocks where a block is composed of 2x2 cells with each cell size of 8x8 pixels. There are three major steps in HOG computation: gradient computation, gradient vote, and normalization.

Assume *f(x,y)* is the intensity map of the input image at position *(x,y)*, the gradients at the *x*- and *y*-directions of each pixel are calculated as:

$$\begin{aligned} g_x(x,y) &= f(x+1,y) - f(x-1,y) \\ g_y(x,y) &= f(x,y+1) - f(x,y-1) \end{aligned}, \quad (1)$$

and the corresponding magnitude and orientation are

$$m(x,y) = \sqrt{g_x(x,y)^2 + g_y(x,y)^2} \quad (2)$$

$$\theta(x,y) = \tan^{-1}\left(\frac{g_y(x,y)}{g_x(x,y)}\right) \quad (3)$$

The computations of magnitude and orientation involves complex arithmetic operations such as square-root and arctangent which makes them difficult for hardware implementations.

After the gradient computation, the angle range of 0° to 180° is divided into 9 bins with 20° in each bin. The magnitude of a pixel is added into a bin according to the pixel's orientation. The accumulated magnitudes of the 9 bins result in a histogram as shown in Fig. 1. In order to reduce the aliasing, the following weightings are applied to bin *i* and its two neighboring bins as follows:

$$\begin{aligned} m_i &= (1-\alpha)\times m(x,y) \\ m_{nearest} &= \alpha \times m(x,y) \end{aligned}, \ \alpha = \frac{\theta(x,y)}{\pi/9} - (i+0.5) \cdot \quad (4)$$

Thus, we can generate a histogram of 9 bins for each cell of 8x8 pixels as shown in Fig. 2 where a 36-component feature vector is used to represent a block.

The final step of HOG is normalization. For a block of 2x2 cells, the feature vector V constructed from the 36 histogram bins is normalized as follows:

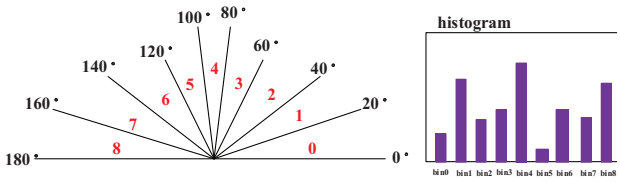

Fig. 1: Histogram of a cell in HOG.

This work is supported by Taiwan's Ministry of Science and Technology under grants MOST 104-2220-E-110-002 and MOST 104-2221-E-110-003-MY2. Electronic Design Automation (EDA) tools and process technologies are supported by Chip Implementation Center (CIC) and Taiwan Semiconductor Manufacturing Company (TSMC)

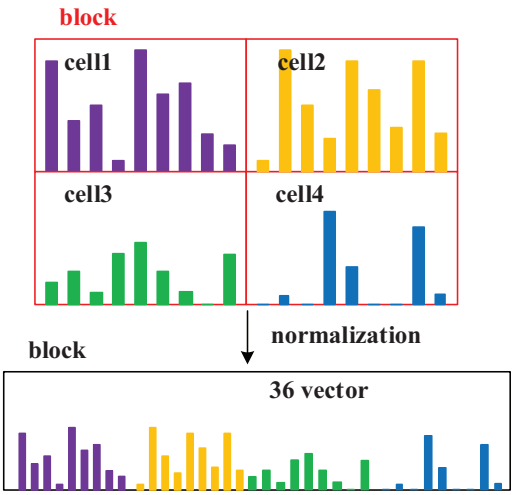

Fig. 2: Histogram normalization of a block in HOG.

$$V_n = \frac{V}{\sqrt{\|V\|^2 + \varepsilon^2}}, \quad \|V\|^2 = \sum_{i=1}^{36} V_i^2 \tag{5}$$

where $V_i, i = 1,2,\cdots,36$ are the 36 accumulated magnitudes from the 4 cells and ε is a small number to prevent division by zero. Note that the computation in Eqn. (5) involves complex operations of square, square-root, and division.

III. Proposed LBP-HOG

In this paper, for a pixel at *(x,y)*, we assign binary values to its four neighboring pixels according to whether the intensity values of the neighboring pixels are larger or smaller. For example, the binary value at position *(x+1,y)* is

$$f'(x+1,y) = \begin{cases} 1 & f(x+1,y) \geq f(x,y) \\ 0 & f(x+1,y) < f(x,y) \end{cases} \tag{6}$$

Binary values $f'(x-1,y)$, $f'(x,y+1)$, $f'(x,y-1)$ of the other three neighbors can be defined similarly. The new gradients can be computed using these binary values:

$$\begin{aligned} g'_x(x,y) &= f'(x+1,y) - f'(x-1,y) \\ g'_y(x,y) &= f'(x,y+1) - f'(x,y-1) \end{aligned}. \tag{7}$$

Thus, from Eqn. (2), the possible magnitude value is either 0, 1, or $\sqrt{2} \approx 1 + 2^{-2} + 2^{-3}$.

Note that we still compute the orientation using Eqns. (2)(3) in order to maintain the same HOG feature size. In this case, the magnitude can be approximated using shift-add operations and comparators as follows:

$$\sqrt{g_x(x,y)^2 + g_y(x,y)^2} \approx \max((0.875 g_{\max} + 0.5 g_{\min}), g_{\max}) \tag{8}$$

where $g_{\max}$ and $g_{\min}$ are the maximum and minimum values of $|g_x(x,y)|$ and $|g_y(x,y)|$ respectively.

The bin number of a pixel can be found by checking which of the following inequalities is satisfied:

Table 1. Approximations of tangent values for bin's boundary angles.

tangent (1st quadrant)	approximation
$\tan(\pi/9)$	$2^{-2} + 2^{-3}$
$\tan(2\pi/9)$	$2^{-1} + 2^{-2} + 2^{-4}$
$\tan(3\pi/9)$	$2^{0} + 2^{-1} + 2^{-2}$
$\tan(4\pi/9)$	$2^{2} + 2^{1} + 2^{-1} + 2^{-3}$

$$g_x(x,y)\tan\theta_i \leq g_y(x,y) \leq g_x(x,y)\tan\theta_{i+1}, \quad i = 0,1,\cdots,8 \tag{9}$$

where $\theta_i, i = 0,1,\cdots,9$ are the 10 boundary angles of the 9 bins. In fact, due to the symmetry of tangent function, we only need to store in a lookup table the 5 tangent values as shown in Table 1 where the tangents of bin boundary angles are further approximated so that the multiplication in Eqn. (9) can be realized using only shift-add operations.

Since the simplified magnitude computation of Eqn. (7) is only approximations, Eqn. (4) of weightings can also be replaced by

$$\begin{aligned} m_i &= 0.5 \times m(x,y) \\ m_{nearest} &= 0.5 \times m(x,y) \end{aligned} \tag{10}$$

which involves only hardwire shift.

In this paper, we also propose to use thresholding and binarization as shown in Fig. 3 to replace the complicate normalization of Eqn. (5). That is, a threshold value is selected to set the histogram values to be either 0 or 1. Compared with the original normalization that requires computations of vector length, square-root, and division, this simplification can significantly reduce the computation and storage hardware with only slight degradation in the pedestrian detection rate. Table 2 compares the detection rate using the original HOG and the proposed simplified HOG for pedestrian detection using support vector machine (SVM) with different sizes of training data sets and test data sets. We observe that there are only slight differences in the detection rates. After the above approximations to simplify hardware design, all the

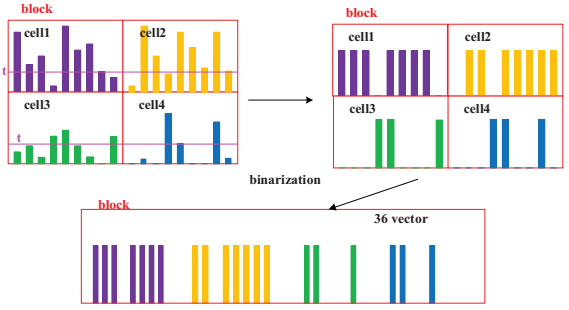

Fig. 3: Proposed binarization that replaces normalization.

Table 2. Comparison of detection rates.

data set size	original HOG	proposed HOG
800	98.5%	93.5%
500	93..0%	88.5%

complicated arithmetic operations are replaced by simple additions/subtractions, hardwired shifts and indexing the small lookup table. Furthermore, the internal buffer size is also reduced significantly.

Fig. 4 is the overall architecture of the proposed simplified HOG design. The line buffer stores scanlines of intensity map *f(x,y)* from the input image. The GLBP performs the operations of Eqn. (7) to be used in LBP-based magnitude computation while the normal gradient performs the original gradient computations of Eqn. (1) for the orientation calculation. Then, the simplified magnitudes and orientation are employed to create the histograms. Finally, the thresholding/binarization generates the binary version of the feature vectors as shown in Fig. 3.

Table 3 compares the area cost of the proposed design with a recent HOG hardware implementation in [8] which we re-implement for the test image size of 64x128 pixels as suggested in [1] for the verification of pedestrian. The synthesis results are based on TSMC 90nm standard cell library. We observe that the proposed design can reduce the area cost due to the simplification of LBP-based magnitude computation and the replacement of normalization by binarization.

Table 4 makes comparison for variants of HOG algorithms. LBP-HOG [9] uses the simplified binary gradients similar to Eqn. (7) for a window and counts the number of transitions, leading to a feature vector with 58 components for a block. In [11], both original HOG and LBP-HOG are combined to improve the detection rate, leading to 66-dimensional feature vectors. Both [9] and [11] only considers the software implementation. The HOG hardware in [8] uses the square-root simplification of Eqn. (8) along with some approximation for the normalization of Eqn. (5). Our proposed design demonstrate a trade-off between hardware cost and detection rate by preventing the complex computation of magnitude and normalization.

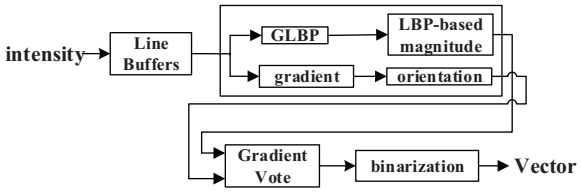

Fig. 4: Overall architecture of the proposed HOG design.

Table 3. Area profiling and comparison of two HOG hardware designs.

Area (um^2)	HOG [8]	proposed HOG
color conversion	300	300
intensity line buffer	17,050	15,489
gradient & magnitude	2,340	2,868
block line buffer	251,135	104,967
gradient vote	397,025	63,932
normalization	553,782	6,440
total	2,396,773	382,116

Table 4. Comparison of various HOG methods.

	original HOG [1]	LBP-HOG [9]	combined HOG [11]	simplified HOG [8]	proposed
SW/HW	-	SW	SW	HW	HW
complexity	high	low	highest	medium	low
features	real	real	real	real	{0,1}
area cost	high	-	-	medium	low
buffer size	high	-	-	medium	small
SVM time	medium	slow	slowest	medium	fast
detection rate	high	low	highest	medium	medium

IV. CONCLUSION

This paper presents an efficient hardware design for HOG which is widely used in many computer vision applications including pedestrian detection. The main concept is to adopt local binary pattern (LBP) to approximate gradient calculation in order to simplify the magnitude computations. Furthermore, the complicate normalization is replaced by binarization, leading a significant reduction in computation hardware area and buffer size. Experimental results show that the proposed design significantly reduces the area cost without too much sacrifice in pedestrian detection rate.

REFERENCES

[1] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 886-893, July 2005.

[2] R. Kadota, et al., "Hardware Architecture for HOG Feature Extraction," *Proc. IEEE Conf. Intelligent Information Hiding Multimedia Signal Processing*, pp. 1330-1333, Nov. 2009.

[3] K. Negi, et al., "Deep Pieplined One-Chip FPGA Implementgation of a Real-Time Image-Based Human Detection Algorithm," *Proc. Intl. Conf. Field Programmable Technology (FPT)*, pp. 1-8, 2011.

[4] S. Lee, *et al*., "HOG Feature Extractor Circuit for Real-time Human and Vehicle Detection," *Proc. TENCON 2012 - 2012 IEEE Region 10 Conference*, pp.1-5, Nov. 2012.

[5] M. Komorkiewicz, M. Kluczewski, and M. Gorgon, "Floating-Point HOG Implementation for Real-Time Multiple Object Detection," *Proc. Intl. Conf. Field Programmable Logic and Applications (FPL)*, pp. 711-714, 2012.

[6] K. Mizumo, et al., "Architectural Study of HOG Feature Extraction Processor for Real-Time Object Detection," Proc. IEEE Workshop on Signal Processing Systems (SiPS), pp. 197-202, 2012.

[7] M. Hemmati, *et al*., "HOG Feature Extractor Hardware Accelerator for Real-time Pedestrian Detection," *Proc. Euromicro Conference on Digital System Design (DSD)*, pp. 543-550, Aug. 2014.

[8] P.-Y. Chen et al., "An Efficient Hardware Implementation of HOG Feature Extraction for Human Detection," *IEEETrans. On Intelligent Transportation Systems*, vol. 15, pp. 656-662, Apr. 2014.

[9] X. Wang, T. X. Han, and S. Yan, "An HOG-LBP Human Detector with Partial Occlusion Handling," *Proc. IEEE Intl. Conf. Computer Vision*, pp. 32-39, 2009.

[10] Ning Jiang et al , A. Serackis, "Gradient Local Binary Patterns for human detection," *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 978-981, May 2013.

[11] G. Gan and J. Cheng, "Pedestrian Detection Based on HOG-LBP Feature," *Proc. Intl. Conf. Computational Intelligence and Security*, pp. 1184-1187, 2011.