CORDIC-based parameters-fusion HOG IP for extracting feature

Aiying Guo^{1,2a)}, Meihua Xu¹, and Feng Ran³

¹ School of Mechatronics Engineering and Automation, Shanghai University,
No. 149 Yanchang RD., Zhabei District, Shanghai City 200072, China

² Department of Electrical and Mechanical Engineering, Shan Xi Vocational and
Technical College of Light Industry,

No. 59 Madaopo RD., Xinghualing District, Taiyuan City 030013, China

³ Microelectronics Research and Development Center, Shanghai University

a) gayshh@shu.edu.cn

Abstract: Histograms of Oriented Gradients (HOG) descriptor significantly outperforms other features for object detection, especially for pedestrian detection. Exploitation of high performance and efficient HOG IP has been a research hotspot in Automotive Electronic. In this letter, a CORDIC-based Parameters-fusion LUT HOG algorithm is proposed and made as the hardware IP (Intelligent Property). To decrease the computational effort and improve the accuracy, pipeline Coordinate Rotation Digital Computer (CORDIC) is applied to generate the magnitude and orientation of gradient. Parameters-fusion Look-Up-Table (LUT) based regional division can operate the tri-linear interpolation in HOG with high speed. Circuit design and chip fabrication were performed using 0.18 μm CMOS technology. Measurement result shows that this CORDIC-based Parameters-fusion LUT HOG IP reduces the hardware overloads and can be used as the general IP for extracting features.

Keywords: CORDIC, pipeline, HOG, chip, IP, extracting feature, object detection, pedestrian detection

Classification: Integrated circuits

References

- [1] N. Dalal and B. Triggs: "Histograms of oriented gradients for human detection," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005 **1** (2005) 886 (DOI: [10.1109/CVPR.2005.177](https://doi.org/10.1109/CVPR.2005.177)).
- [2] P. Dollar, *et al.*: "Pedestrian detection: an evaluation of the state of the art," IEEE Trans. Pattern Anal. Mach. Intell. **34** (2012) 743 (DOI: [10.1109/TPAMI.2011.155](https://doi.org/10.1109/TPAMI.2011.155)).
- [3] P. F. Felzenszwalb, *et al.*: "Object detection with discriminatively trained part-based models," IEEE Trans. Pattern Anal. Mach. Intell. **32** (2010) 1627 (DOI: [10.1109/TPAMI.2009.167](https://doi.org/10.1109/TPAMI.2009.167)).
- [4] D. L. Cosmo, *et al.*: "Pedestrian detection system based on HOG and a modified version of CSS," Proc. SPIE **9445** (2015) 94450I (DOI: [10.1117/12.](https://doi.org/10.1117/12.)

- 2180766).
- [5] R. Benenson, *et al.*: “Ten years of pedestrian detection, what have we learned?,” Computer Vision-ECCV 2014 Workshops (2014) 613 (DOI: [10.1007/978-3-319-16181-5_47](https://doi.org/10.1007/978-3-319-16181-5_47)).
 - [6] C. S. Park and S. Park: “CSD-based CORDIC algorithm and its VLSI implementation,” IEICE Electron. Express **13** (2016) 20160393 (DOI: [10.1587/elex.13.20160393](https://doi.org/10.1587/elex.13.20160393)).
 - [7] P. K. Meher, *et al.*: “50 years of CORDIC: algorithms, architectures, and applications,” IEEE Trans. Circuits Syst. I, Reg. Papers **56** (2009) 1893 (DOI: [10.1109/TCSI.2009.2025803](https://doi.org/10.1109/TCSI.2009.2025803)).
 - [8] L. I. Jun-qiang, *et al.*: “32 × 32 high-speed multiplier design and implementation,” Microelectron. Comput. **26** (2009) 23.
 - [9] K. Negi, *et al.*: “Deep pipelined one-chip FPGA implementation of a real-time image-based human detection algorithm,” International Conference on Field-Programmable Technology, FPT (2011) 1 (DOI: [10.1109/FPT.2011.6132679](https://doi.org/10.1109/FPT.2011.6132679)).
 - [10] <http://pascal.inrialpes.fr/data/human/>

1 Introduction

Histograms of Oriented Gradients (HOG) are the classic feature descriptor based on the image’s gradient magnitude and orientation in object detection or video surveillance [1]. HOG also can be used as the basic cell in many algorithms for feature extraction, such as Deformable Parts Module (DPM), Color Self Similarity (CSS) and so on [2, 3, 4]. The merit is that HOG can vividly describe the local gradient or edge orientation for the detector image. But the huge amount of calculation is the bottleneck for practical chip implementation (1). One of the attempts made to break the bottleneck is the so-called LUT (Look-Up-Table) HOG algorithm, which makes HOG IP (Intelligent Property) possible to huge size [5]. At the same time, the focus of attention in the design of HOG is merely the achieved accuracy. Nevertheless, the speed is a key requirement in the realistic scenarios for real-time image-based detection. Few embedded detection works are present in the literature. This motivates and encourages the research on hardware acceleration. So far, hardware implementation of HOG-based detection, such as pedestrian detection, has been investigated.

In order to reduce the HOG calculation and be used as a general hardware IP for the embed system, a CORDIC-based Parameters-fusion LUT HOG algorithm is proposed and designed as one chip. In this letter, pipeline Coordinate Rotation Digital Computer (CORDIC) is adopted to calculate the orientation and magnitude of gradient for promoting the speed; LUT for location and orientation parameters based on the regional division is applied to operate tri-linear interpolation in HOG, namely Parameters-fusion LUT [4]. Software version and its hardware IP are both evaluated. Evaluation results show that this HOG decrease the calculation amount and can be used as the general IP.

2 Theory of HOG algorithm

The HOG uses the local histograms of oriented gradients from pixel luminance to extract features. The HOG decomposes an image into small cells, computes a 1D

histogram of oriented gradients in each cell, normalizes the results with a block-wise pattern, and returns a vector. Then, stacking the vector block by block can be used as a detector window HOG features for object detection, for example by means of a Support Vector Machine (SVM).

As shown in Fig. 1, HOG features in one detector window are generated by 1D histogram using tri-linear interpolation based on location, orientation and magnitude of gradient.

Fig. 1. Block diagram of the HOG algorithm

The process of HOG descriptor extraction consists of seven stages. Details of each stage are explained in the following.

Stage 1: Calculate luminance value in a given image/frame. The luminance value H for each pixel is calculated based on the RGB values according to Eq. (1).

$$H = (R + G + B)/3 \quad (1)$$

Let $H(x, y)$ stands for the luminance value of the pixel at the coordinate (x, y) , while H , R , G and B takes the value from 0 to 255.

Stage 2: Compute horizontal and vertical gradient with no smoothing. The horizontal gradients $G_x(x, y)$ and vertical gradients $G_y(x, y)$ are defined by:

$$G_x(x, y) = H(x + 1, y) - H(x - 1, y) \quad \forall(x, y) \quad (2)$$

$$G_y(x, y) = H(x, y + 1) - H(x, y - 1) \quad \forall(x, y) \quad (3)$$

Stage 3: Compute magnitude and orientation of gradient. In detail, orientation $\theta(x, y)$ and magnitude $G(x, y)$ can be generated by:

$$\theta(x, y) = \tan^{-1} \left(\frac{G_y(x, y)}{G_x(x, y)} \right) \quad (4)$$

$$G(x, y) = \sqrt{G_x(x, y)^2 + G_y(x, y)^2} \quad (5)$$

Stage 4: Describe the cell, block and bin. Divide a detector window into 16×16 blocks with 50% overlap in both vertical and horizontal direction. Each block should consist of 2×2 cells in one block. Non-direction bin is adopted in HOG, while $[0^\circ, 180^\circ]$ is divided into 9 bins [1].

Stage 5: Orientation/Spatial Binning. This operation can calculate a weighted vote for an orientation histogram channel based on the orientation of the gradient element, pixel by pixel. The votes are accumulated into orientation bins over local spatiality region in every cell, as Fig. 2 shows. As Fig. 2 shows, the magnitude of one pixel, such as the pixel (x, y) , makes different contributions to four cells in one block.

Fig. 2. Tri-linear interpolation in one block and 1D histograms in one cell

Let $f(x, y, \theta_i)$ stands for the histogram of gradient in the i bin, while θ_i is the intermediate orientation value in the i bin. 1D histogram can be operated by:

$$f(x, y, \theta_i) \leftarrow f(x, y, \theta_i) + G(x, y)\theta_p(x, y, \theta_i)P_p(x, y), \quad i = 1, 2, 3, 4, 5, 6, 7, 8, 9 \quad (6)$$

$$\theta_p(x, y) = 1 - \frac{\theta(x, y) - \theta_j}{d\theta}, \quad j = 1, 2, \theta(x, y) \in [0^\circ, 180^\circ) \quad (7)$$

$$P_p(x, y) = \left(1 - \frac{(x - x_j)}{dx}\right) \left(1 - \frac{(y - y_j)}{dy}\right), \quad j = 1, 2 \quad (8)$$

Stage 6: Normalize all features in one block with one reference value. Because of the local variations in illumination and foreground-background, the variation of gradient values is very large. Then, the normalization of gradient can decrease this impact. The methods of normalization are operated with L0-Norm, L1-Norm and L2-Norm. L2-Norm or L1-Norm is adopted in HOG. The value of ε is small to avoid the case that the denominator may be zero.

$$v = \frac{v_k}{\sqrt{\|v_k\|^2 + \varepsilon}} \quad \text{or} \quad v = \frac{v_k}{(\|v_k\|_1 + \varepsilon)} \quad (9)$$

Stage 7: Concatenate the all HOG features in one detector window.

Commonly, the detector window size is 64×128 pixels. So, a vector of HOG feature is 3780, which is extracted from each detector window.

3 Analysis of HOG

The calculation process of the HOG descriptor, which is discussed in the previous section, is still too complicated to be realized as the hardware IP. So, in order to reduce the calculation complexity and achieve higher performance, this letter adopts three methods to improve the HOG algorithm.

First approach is approximation. The approximation is used to replace the square root to solve small values of gradients in stage 2 in section 2. When the values of $G_x(x, y)$ and $G_y(x, y)$ are smaller, the magnitude can be generated by Eq. (10) through approximation. The threshold of magnitude is 5.

$$G(x, y) = \sqrt{G_x(x, y)^2 + G_y(x, y)^2} \approx |G_x(x, y)| + |G_y(x, y)| \quad (10)$$

Second algorithm is that pipeline CORDIC calculates the magnitude and orientation of the gradient [5, 6]. When the value is larger than the threshold value, the square root will be operated with CORDIC algorithm, not by approximation of gradient. At the meantime, orientation of gradient also can be generated by CORDIC, which can be seen as a set of shift-and-add iteration operations to obtain the orientation [6].

Suppose that a vector $f_0 = [x_0, y_0]^T$ is rotated by an angle θ to get another vector $f_N = [x_N, y_N]^T$, as Eq. (11) showed:

$$\begin{pmatrix} x_N \\ y_N \end{pmatrix} = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = R \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \quad (11)$$

R is the rotation matrix.

To achieve simplicity of hardware realization when rotating, the key ideas used in CORDIC arithmetic are:

- 1) Decompose the rotations into a sequence of elementary rotations through predefined angles that could be implemented with minimum hardware cost.
- 2) To avoid scaling, that might involve square-root or division operations.

So, the rotation angle θ is then decomposed into the N sub-angles of

$$\begin{cases} \theta = \sum_{n=0}^N d_n \theta_n \\ \theta_n = \arctan\left(\frac{1}{2^n}\right) \end{cases} \quad (12)$$

In Eq. (11), $d_n \in \{+1, -1\}$ is given as:

$$d_n = \text{sign}\left(\theta - \sum_{k=0}^{n-1} d_k \cdot \arctan(2^{-k})\right) \quad (13)$$

Then, if the i -th rotation can be expressed as $f_i = R_{i-1} f_{i-1}$, then the rotation matrix R_i can be defined as:

$$R_i = \begin{pmatrix} \cos \theta_i & -\sin \theta_i \\ \sin \theta_i & \cos \theta_i \end{pmatrix} = \cos \theta_i \begin{pmatrix} 1 & -\tan \theta_i \\ \tan \theta_i & 1 \end{pmatrix} = K_i \begin{pmatrix} 1 & -d_i 2^{-i} \\ d_i 2^{-i} & 1 \end{pmatrix} \quad (14)$$

K_i is the scale factor. In order to avoid scaling in the CORDIC algorithm, the all K_i can be handled alone and redesigned as:

$$K = \prod_{i=0}^{n-1} K_i \approx 0.6237 \quad (15)$$

Based on the Eq. (14), the value of K is ordinary used as 0.6237 [7]. Then, the i -th iteration in CORDIC algorithm is re-expressed as:

$$\begin{cases} x_{i+1} = x_i - d_i 2^{-i} y_i \\ y_{i+1} = y_i + d_i 2^{-i} x_i \\ \theta_{i+1} = \theta_i - d_i \arctan(2^{-i}) \end{cases} \quad (16)$$

The number of iterations and accuracy are in the direct ration, while grows in inverse proportion to the processing speed. In this HOG IP, the number of iteration

is 12 to make sure the accuracy of results, while the 11 stage pipelines are inserted into the CORDIC algorithm to promote processing speed.

The calculation of tri-linear interpolation can consume about more than 70% of the total feature extraction time. Then, in order to reduce the processing time, the tri-linear interpolation is revised with new method, called Parameters-fusion LUT. The strategy of Parameters-fusion LUT is decomposed into four steps.

First-step: regional division in one cell and block. Many experiments show that the aim of tri-linear interpolation operations in HOG is to eliminate the regional mixing for HOG features in adjacent cells. But the fact is that the pixel in one cell makes different contributions to the other three cells in one block according to their coordinate. So, every cell in one block can be divided into 2 parts based on their coordinate as Fig. 3 shows, which one part is filled with green $G(x, y)$ and the other is filled with red $R(x, y)$. The magnitudes in green region just make contributions to their own cell, and the red make contributions to all cells in one block.

Fig. 3. Different regional division in one cell and block

Based on these analyses, the Eq. (9) is changed to:

$$P_p(x, y) = \begin{cases} \left(1 - \frac{(x - x_i)}{dx}\right) \left(1 - \frac{(y - y_i)}{dy}\right), & i = 1, 2, (x, y) \in R(x, y) \\ 1, & (x, y) \in G(x, y) \end{cases} \quad (17)$$

Second-step: Eq. (7) is rearranged to reduce the calculation complexity. Eq. (7) shows that the magnitude of one gradient will be voted to two bins according to the orientation from the luminance value of one pixel. In this HOG, these magnitudes are only projected into their own bins, which only make contribution to one bin. So, the Eq. (7) can be rearranged:

$$\theta_p(x, y) = 1, \theta_p(x, y) \in \theta_i \quad (18)$$

At the meantime, Eq. (6) can be changed:

$$\begin{aligned} f(x, y, \theta) &\leftarrow f(x, y, \theta) + G(x, y) \theta_p(x, y, \theta) P_p(x, y) \\ &\leftarrow f(x, y, \theta) + G(x, y) P(x, y) \end{aligned} \quad (19)$$

Third-step: construction LUT. Based on Fig. 3, the total number of parameters in one cell is 64. These parameters can be reusable in other three cells in the same block. Then, in order to decrease the processing time, the registers as LUT are used to store these parameters. Compared with the size and speed, the register array not SRAM is the perfect device to complete this operation.

$$P(x, y) = \begin{cases} P, (x, y) \in R(x, y) \\ 1, (x, y) \in G(x, y) \end{cases} \quad (20)$$

The multiplicand is also used in Eq. (19). So, the Booth-32 bit multiplicand is used to calculate 1D histogram in one block [8].

The operation of first, second and step is called as Parameters-fusion method.

The last step is the normalization. The strategy of binaryzation is used in this HOG to replace the normalization [9]. In detail, if the L2-Norm is used in HOG IP, 36 divisions, 1 square root, 36 additions and 36 squares are paid up to operate the normalization. The normalization is used to remove the illumination effect to the HOG features. Through the normalization operation, the HOG features are pulled up to the same reference values. Then, the binaryzation of HOG features can give the same function compared with the traditionally normalization, while the binaryzation can be realized more simply than the L2-Norm or some other normalization methods. So, the binaryzation is adopted in this HOG IP. Sometimes, the different HOG features before normalization will be level to the designer adjust their methods. Then, the normalization operation of our HOG IP will be transformed to the board.

4 Simulation results

In this section, the performance of CORDIC-based Parameters-fusion LUT HOG in software is quantity by Detection Error Tradeoff (DET) curves with the statistical data from the results.

Before evaluation, the INRIA dataset must be prepared [10]. The INRIA dataset supplies test and train images of real-world scenes for evaluating the function of algorithms. In this evaluation, 3,000 images were used for offline machine learning, while 14,000 images were used as the evaluation data. The size of training and test images are 64×128 pixels.

The performance of this HOG is evaluated with two parts.

First part is the whole performance evaluation of this HOG algorithm. CORDIC-based Parameters-fusion LUT HOG is improved based on the HOG. Systems for pedestrian detection, HOG+Linear SVM and this HOG+Linear SVM, are compared to each other for distinguishing their functions. The method for quantify is to report miss rates versus false positives per window, calling DET curves of different algorithms. Lower values are better. As the Fig. 4 shows, the DET curves from HOG+Linear SVM and Our HOG+Linear SVM are used to quality algorithm performance. As Fig. 4(a) shows, Our HOG is equivalent to the Dalal's HOG at 10^{-4} FPPW. So, the CORDIC-based Parameters-fusion LUT HOG can be used to extract image's feature without performance sacrifice.

Next step is to evaluate the performance this HOG with different numbers of stages in CORDIC. Fig. 4(b) shows the impact with the different number of stages on the performance using DET curve. As Fig. 4(b) shows that, when the number of stage in CORDIC is bigger than 12, the performance has minor improvement. This minor improvement isn't just no-worth with the sacrifice of size. So, the CORDIC algorithm will operate 12 iterations to complete the calculation of magnitude and orientation.

(a)

(b)

Fig. 4. (a) DET curves of two algorithms (b) DET curves with different iteration

5 VLSI implementation results and chip measurement

Fig. 5 shows the block diagram of this HOG structure using CORDIC and LUT. To evaluate the hardware overheads, our HOG IP architecture is firstly designed by Verilog HDL in RTL level.

Fig. 5. Block diagram of improved HOG IP

To investigate the benefits of the proposed HOG IP, typical HOG IP is also designed by Verilog HDL in RTL level. Table I and Table II show the hardware overheads details with the Quartus II using Cyclone III. Analysis the Table I and II, our HOG IP can decrease about 12% overheads than the HOG IP. In order to simplify the implementation, fixed-point arithmetic with a 8-bit fraction part is used in this HOG IP.

Table I. FPGA synthesis result report of our HOG IP

Slice logic	Used	Utilization (%)
Number of slice registers	3351	22
Total Combination function	6313	41

Table II. Traditionally HOG FPGA report

Slice logic	Used	Utilization (%)
Number of slice registers	4657	29
Total Combination function	8156	53
Total Memory bits	16384	3

At the meantime, a 180-nm CMOS standard-cell library from Global Foundry Technology is used to synthesis this HOG IP. In fact, CORDIC-based Parameters fusion HOG IP contains CONTROL module to determine whether all pixels in one detector window are injected into IP or not. If the number of input data isn't 8192, the CONTROL module will give the STOP signal to the other modules in HOG, and generate the STOP signal, which has 3 serial pluses, and reset all modules, calling ABNORMAL mode as Fig. 6 shown. The other mode is NORMAL.

Fig. 6. Abnormal mode of improved HOG IP

Fig. 7. Input signals of improved HOG IP

In NORMAL mode, the data.in and ready.in are exhibited in Fig. 7. Fig. 8 shows the signals of bin.out and ready.out in NORMAL mode.

Fig. 8. Output signals of improved HOG IP

Additionally, this HOG IP is developed with ceramic leaded chip carrier (CLCC 44), which is labeled by SHU-IRAC18. The SHU-IRAC18 chip contains one test chain using the technology of design for test (DFT). Simple coding is designed to test whether SRAM is right or not, and don't has the repaired function.

Fig. 9 shows die and chip of IP, which size is about 2.5 mm × 2.5 mm. The performance of the manufactured chip was tested along with the testing hardware platform using SHU-IRAC18 for pedestrian detection.

Fig. 9. Die and chip of IP

The testing platform can be used for object detection, especially for pedestrian detection as Fig. 10 shows.

Fig. 10. Testing platform for SHU-IRAC18

Fig. 11 shows the examples of pedestrian detection using SHU-IRAC18.

Fig. 11. Examples of pedestrian detection using SHU-IRAC18

6 Conclusion

This letter shows the theory about the CORDIC-based Parameters-fusion LUT HOG algorithm and the structure of hardware IP. Based on the analysis of CORDIC and HOG algorithm, it is shown that this HOG IP can be reviewed to decrease the hardware overload and improve the speed. Moreover, the VLSI implementation results and chip measurement show that this HOG IP can be used as the general IP to extract features from the images or frames in object detection, especially in pedestrian detection.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Grant: 61376028).