VIETNAM NATIONAL UNIVERSITY, HANOI

UNIVERSITY OF ENGINEERING & TECHNOLOGY

SIS Laboratory, VLSI Systems Design Group

LSI Contest 2017 Report

Human Detection by Histogram of Oriented Gradients

GENERAL INFORMATION

AUTHOR

- Name: Huy Hung Ho
- Class: K59ĐB – UET-VNU
- Email: heroh396@gmail.com
- Mobile: 01642 792 422

RESEARCH TITLE

- **LSI Contest 2017**
- Research description: The design LSI (Large Scale Integration) circuit contest
- Context: Human Detection by Histogram of Oriented Gradients
- Objective: Design the architecture of LSI circuit
- Obtained results (briefly, should be updated frequently): design harware architecture
- References: http://www.lsi-contest.com
- Start: 15th November 2016
- Report deadline: 27th January 2017

VIETNAM NATIONAL UNIVERSITY, HANOI

UNIVERSITY OF ENGINEERING & TECHNOLOGY

SIS Laboratory, VLSI Systems Design Group

CONTENTS

Table of Contents

1 Purpose 3
2 Algorithm 4
2.1 Introduction 4
2.2 HOG 5
2.3 SVM 8
3 Architecture 9
3.1 General Architecture 9
3.2 Cell Input Generation 10
3.3 Cell Histogram Generation 11
3.4 Histogram Normalization 11
3.5 SVM Classification 12
3.6 Processing 12
4 Originality point 14
4.1 Scan image by shift register 14
4.2 Using LUT (Look Up Table) when calculating magnitude gradient 14
4.3 Calculating SVM Classification by combined serial and parallel 15
5 *The* display of a simulation 17
6 Other 17
6.1 Acknowledge 17
6.2 References 17

1 Purpose

- My purpose is can simulation hardware of this project by HDL code.
- Currently, I had designed hardware architecture and simulation it in C code.
- The most important in my architecture is optimized performance and timing.

2 Algorithm

2.1 Introduction

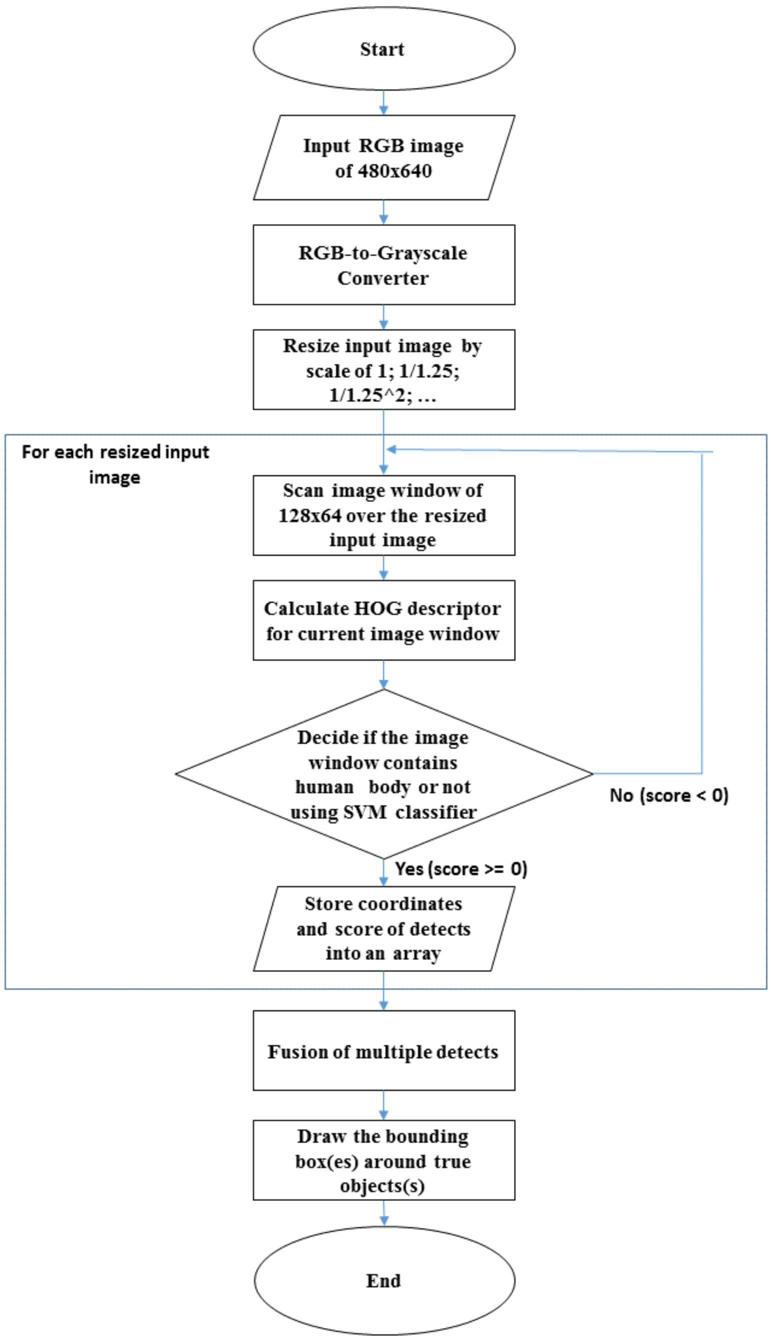
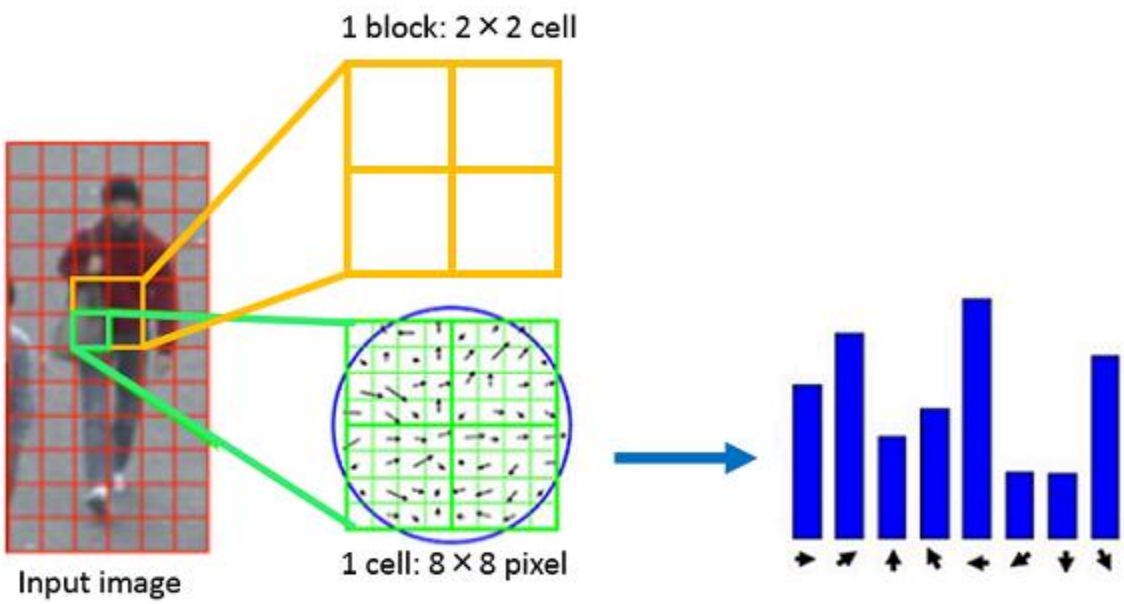


Figure 1: Flowchart of template matching

2.2 HOG

Histogram of Oriented Gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. We can get quantity of strong characteristic from the shape change of the object by dividing a local domain into plural blocks, and making the incline of each the histogram.



The following (1) - (5) is the process to get quantity of HOG.

(1) RGB-to-Gray Converter

Since the detection process using a color image is difficult, the input image is converted into grayscale image. We use the NTSC Coef method to grayscale of the image. The NTSC Coef. Method, in one method of converting from a color image to 256-level grayscale image, it can be calculated by the following equation.

$$I = 0.298912 * \mathrm{R} + 0.586611 * \mathrm{G} + 0.114478 * \mathrm{B}$$

(2) Resize image

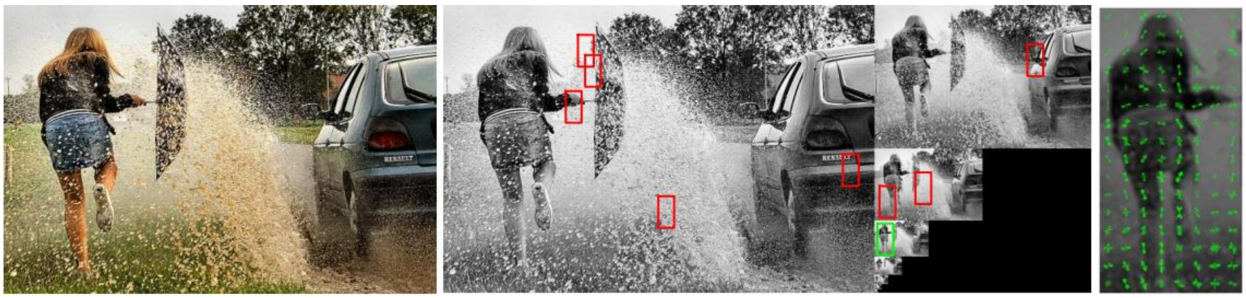


Figure 2: …

The size of a persion in the image depends on the distance from the camera to the persion, then we need resize image with other scale.

(3) Scan window image

Resize grayscale input image at different scales. In our algorithm, the detection process uses an image window of **fixed-size (128 x 64 pixels)** to scan over the whole input image, and the size of the human body in each image varies at different scale. Therefore, algorithm needs to sequentially resize the original-size input image into images with smaller sizes (as long as the image window can be included in the resized input image).

(4) Calculate gradient magnitude and gradient angle from the brightness of each pixel.

The derivative at a pixel is the change of the brighness at this pixel, so we can consider this change by the brighness different of neighboring pixel. Approximate the two components $fx(x,y)$ and $fy(x,y)$ of the gradient of $I(x,y)$ by central differences:

$$fx(x,y) = I(x+1,y) - I(x-1,y)$$

$$fy(x,y) = I(x,y+1) - I(x,y-1)$$

We calculated gradient magnitude and the gradient angle by the following expressions.

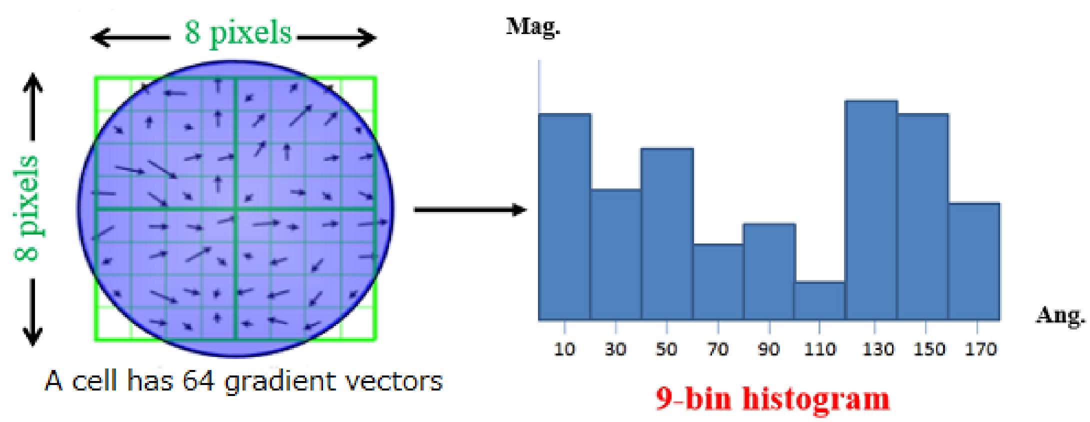
Gradient magnitude:

$$\mathrm{m(x,y)} = \sqrt{\mathrm{fx(x,y)}^2 + \mathrm{fy(x,y)}^2}$$

Gradient angle:

$$\theta(x,y) = \arctan(\frac{fy(x,y)}{fx(x,y)})$$

(5) Calculate the histogram of gradient in each cell.



(6) Figure 3: Histogram of 9 directions

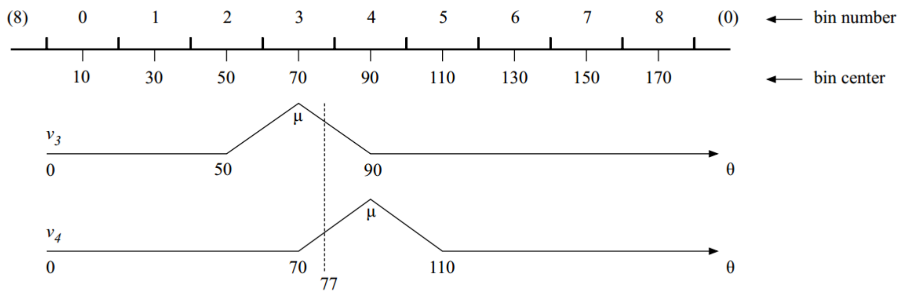
We quantized gradient angle each 20 degrees from zero degree to 180 degrees in 9 directions and calculate histogram in each cell domain.

Divide the window into adjacent, non-overlapping cells of size CxC pixels (C = 8). In each cell, compute a histogram of the gradient orientations binned into B bins (B = 9).

With so few bins, a pixel whose orientation is close to a bin boundary might end up contributing to a different bin, were the image to change slightly. To prevent these quantization artifacts, each pixel in a cell contributes to two adjacent bins (modulo B) a fraction of the pixel's gradient magnitude $m(x, y)$ that decreases linearly with the distance of that pixel's gradient orientation from the two bin centers.

Specifically, the bins are numbered 0 through $B - 1$ and have width $\omega = \frac{180}{B}$. Bin i has boundaries $[\omega(i), \omega(i + 1))$ and center $c(i) = \omega(i + \frac{1}{2})$. A pixel with magnitud $m(x, y)$e and orientation $\theta(x, y)$ contributes a vote:

- To bin number $j = \left\lfloor \frac{\theta}{\omega} - \frac{1}{2} \right\rfloor mod\ B$: $\quad v(j) = m(x, y) \frac{c(j+1) - \theta(x,y)}{\omega}$
- To bin number $(j + 1)\ mod\ B$: $\quad v(j + 1) = m(x, y) \frac{\theta(x,y)\ c(j)}{\omega}$



(6) Normalize each block.

We normalized each block by the following expressions

$$v(i) = \frac{v(i)}{\sqrt{\sum_{k=1}^{q*q*N} v(k)^2 + \varepsilon}}$$

- ε is constant to remove the divisin with zero.
- $\boldsymbol{v}(\boldsymbol{n})$ is gradient angle histogram, q is cell size, N is the number of gradient angle. In addition, the denominator is a grand total of the quantity of HOG characteristic included in 1 block (qXq cell). For one cell, we normalize multiple times.

Block normalization is a compromise: On one hand, cell histograms need to be normalized to reduce the effect of changes in contrast between images of the same object.

On the other hand, overall gradient magnitude does carry some information, and normalization over a block—a region greater than a single cell—preserves some of this information, namely, the relative magnitudes of gradients in cells within the same block.

Since each cell is covered by up to four blocks, each histogram is represented up to four times with up to four different normalizations.

(7) Integrate all histogram.

We find out the matrix HOG of window image

$$HOG\ Feature = [v(1);\ v(2);\ v(3) \dots v(N)]$$

2.3 SVM

Support Vector Machine (SVM) is one of the pattern identification technique and is used for image recognition and speech recognition. We used linear SVM.

Let learn learning data of INRIA Person Dataset, and chooses hyperplane becoming margin maximum in hyperplane dividing into the data of the 2 level class. As a free library of linear SVM, We used LIBLINEAR.

The classifier of person or not person by 2 classes is shown in Figure 4:

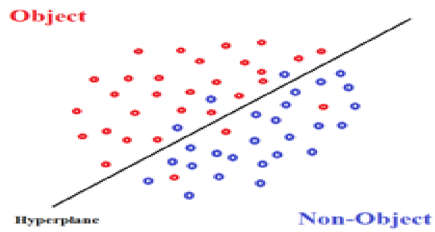


Figure 4: Figure of identification by the two classes classification

Using convolution to determine whether the object in window image:

- Input

$$SVM\ Trained = [u(1);\ u(2);\ u(3) \ldots u(N)]$$

$$HOG\ Feature = [v(1);\ v(2);\ v(3) \ldots v(N)]$$

- Calculating

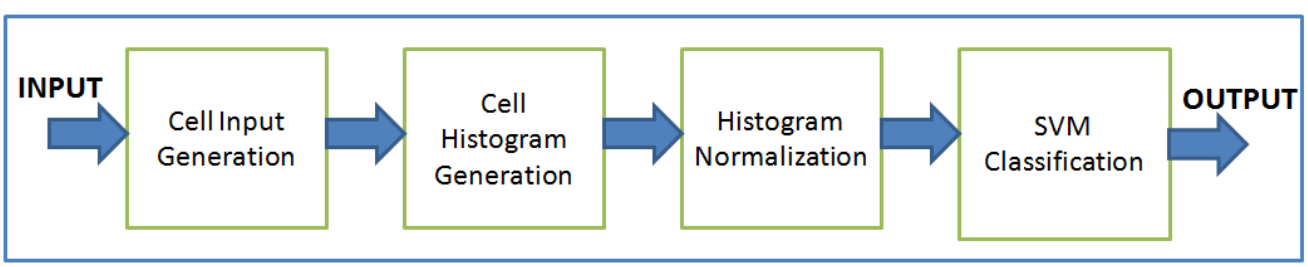
$$Score = \sqrt{\sum_{k=1}^{N} u(k) * v(k)}$$

$$if\ (Score \geq 0)\ then\ detected$$

3 Architecture

3.1 General Architecture

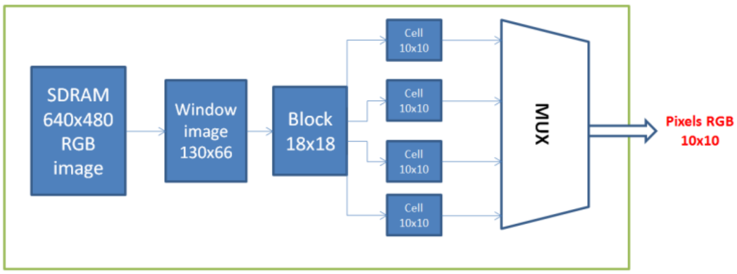
Algorithm is not complete, we must supplement **resize block** and **fusion multiple block**.



General Architecture

Figure 5: Raster scan each cell in row

3.2 Cell Input Generation

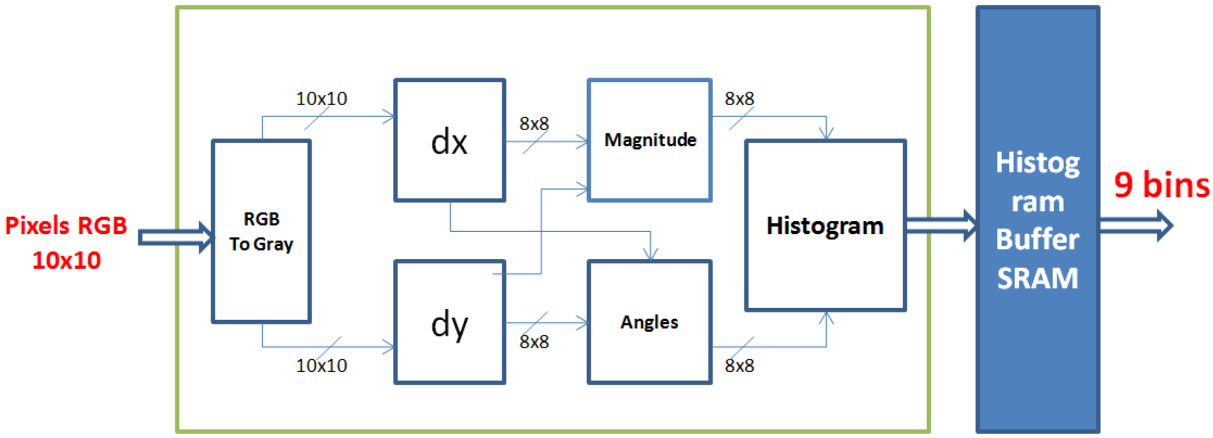


Cell Input Generation

The smallest unit to calculate is a cell. So we use a Multiplexer to turn select the cells of the block.

Because calculating border pixel derivative we must have neighboring pixell, then each cell size 10x10 pixel.

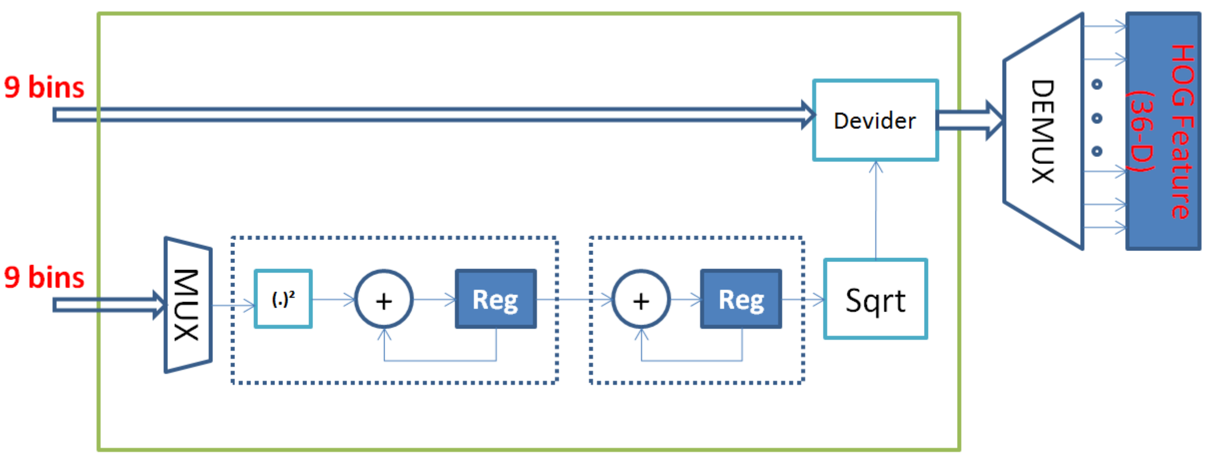
3.3 Cell Histogram Generation



Cell Histogram Generation

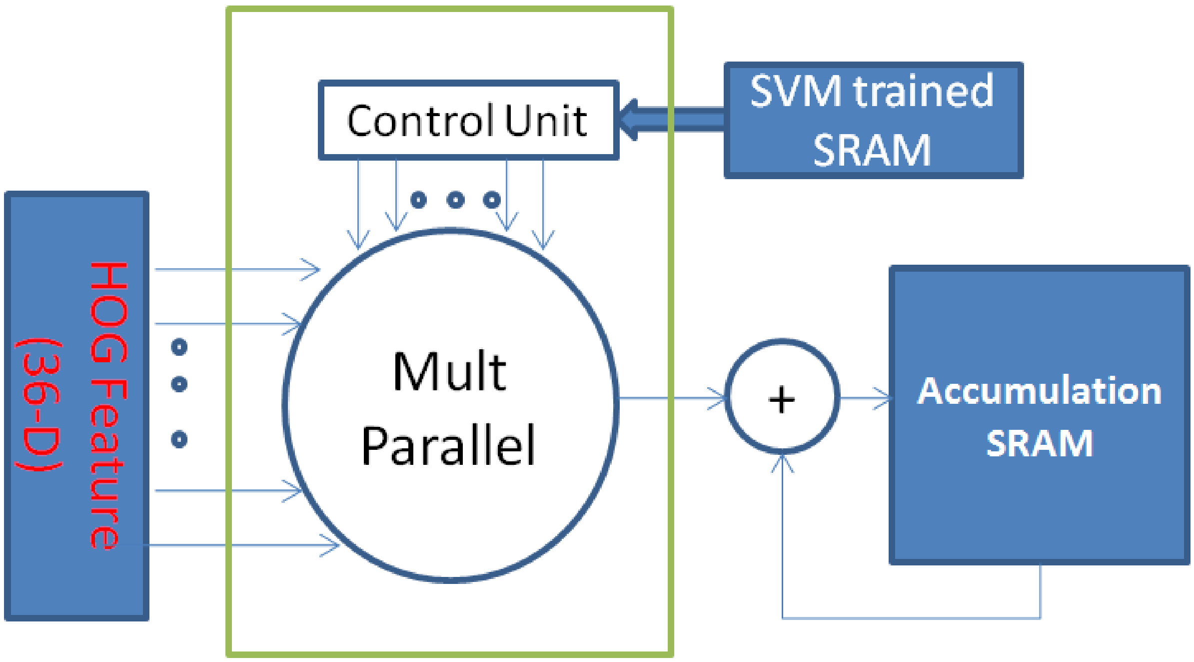
Each cell requires its neighboring cells create four overlapped blocks for normalization. Accordingly, the 9-bin cell histogram must be stored in a buffer SRAM so that it can be used to compute the normalized histogram with respect to the different blocks

3.4 Histogram Normalization



Histogram Normalization

3.5 SVM Classification



©

SVM Classification

3.6 Processing

- The 1st processing, I calculate HOG Cell follow previous flowcharts (from **3.1** to **3.5)**
- Since 2nd processing, I can sharing cell across overlapped window:
  The HOG feature of each cell is immediately used for classication once it is extracted so that it is never buffered or recomputed. This reduces on-chip memory requirements and external memory bandwidth as each pixel is only read once from the off-chip frame buffer.All calculations that require the HOG feature must be completed before it is thrown away.

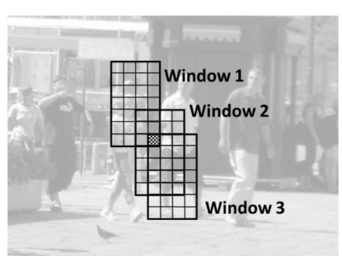
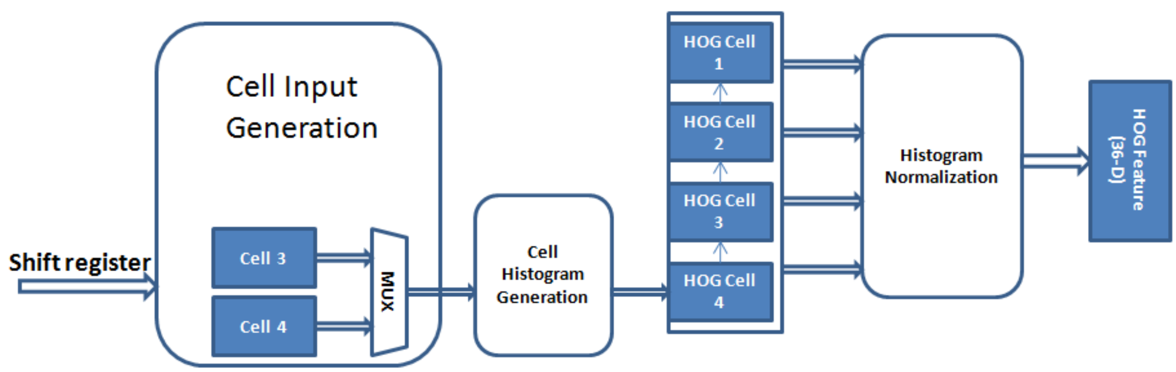


Figure 6: Sharing cell across overlapped windows

Figure 6 shows a simple example, with a small detection window size of 8x8 cells, of how several detection windows share a cell. Each cell effectively appears, and must be accumulated, at all positions in overlapped detection windows. For a 128x64 pixels detection window, each cell is shared with (8x8 = 64) windows, but at different positions within each window.

Figure 7: Sharing cell feature and window feature

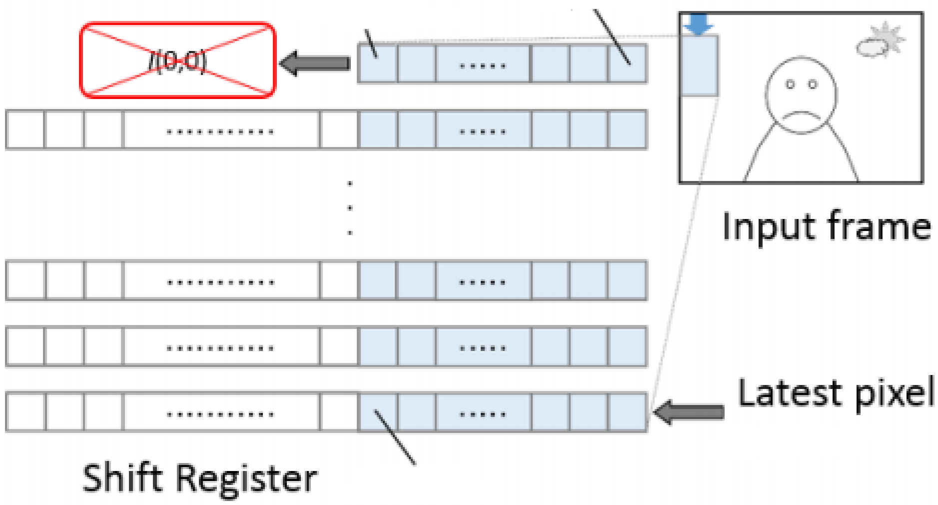
- Description sharing cell in hardware architecture:



4 Originality point

I found 3 point to optimization:

4.1 Scan image by shift register



Instead of scan window image and block image as usual, I shift register 8 pixel to pass next window.

4.2 Using LUT (Look Up Table) when calculating magnitude gradient

- When calculating magnitude gradient, I realize derivative $fx(x,y)$ and derivative $fy(x,y)$ usually valued in the range of 0 to 15.
  In expression:

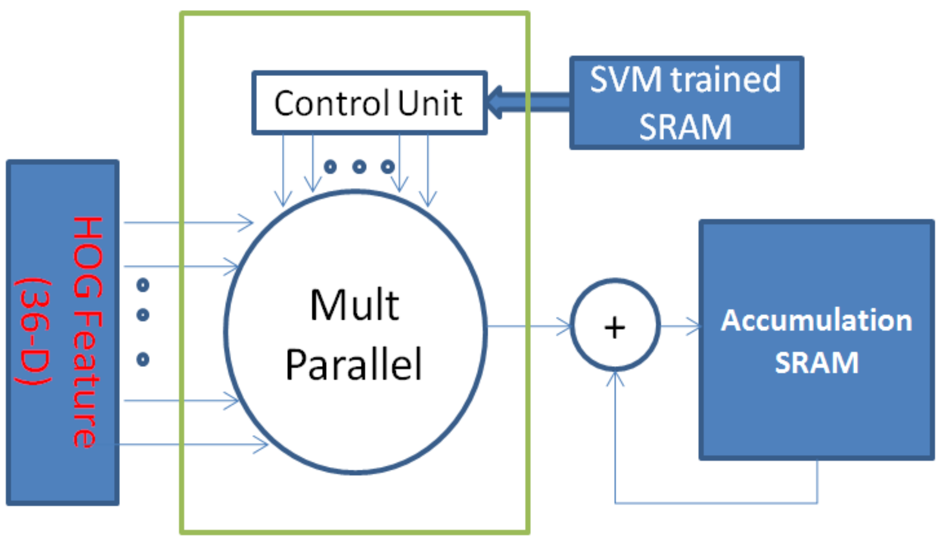
$$m(x,y) = \sqrt{fx(x,y)^2 + fy(x,y)^2}$$

  I use LUT to quickly calculate $fx(x,y)^2$, $fy(x,y)^2$ and I do not use a multiplier, increase calculting speed.

- Pseudo code:

if $fx(x,y) < 16$ *then* $fx(x,y)^2 = LUT\ (fx(x,y))$ *else* $fx(x,y)^2 = fx(x,y) * fx(x,y)$	*if* $fy(x,y) < 16$ *then* $fy(x,y)^2 = LUT(fy(x,y))$ *else* $fy(x,y)^2 = fy(x,y) * fy(x,y)$

4.3 Calculating SVM Classification by combined serial and parallel

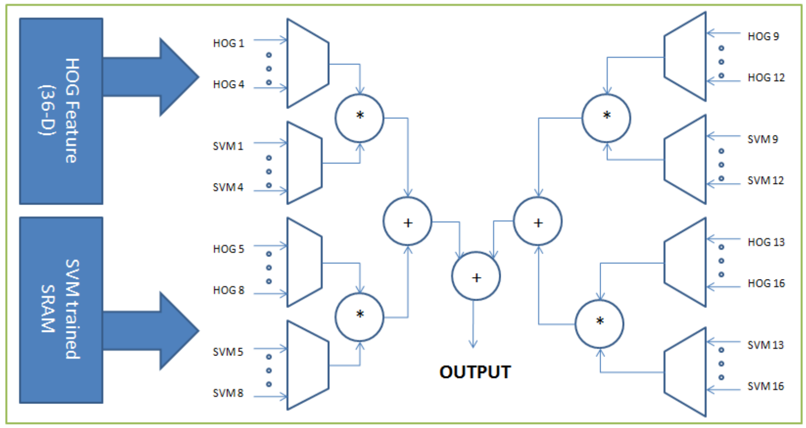


SVM Classification

This image show in the SVM classification. The Mult Parallel block calculate the convolution between block HOG (36-D) and SVM trained (36-D). The sum of convolution save in SRAM buffer.

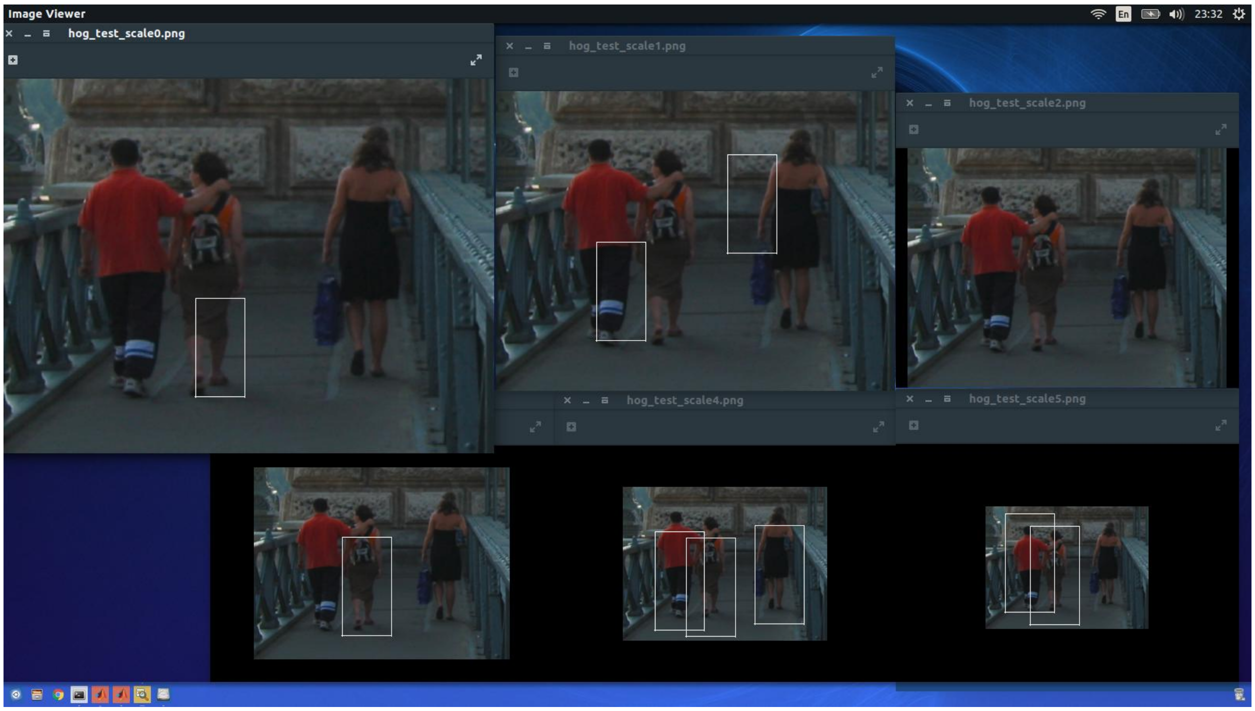
- We use combined serial and parallel to design Mult Parallel block. Using 4 multiplier, so we balanced between timing and performance.

/*TODO: ascertainment */



Mult Parallel

5 The display of a simulation



6 Other

6.1 Acknowledge

I sincerely thank you for my SISLAB (Laboratory for Smart Integrated Systems), especially Tu Teacher and Sinh-Ngoc Nguyen for supporting me in this project.

I do not have enough time to finish this project, but I will try my best if you give me one more chance.

6.2 References

[1] N. Dalal, et al., "Histograms of Oriented Gradients for Human Detection," in Proceedings of the 2005 International Conference on Computer Vision and Pattern Recognition, vol. 2. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893.
[2] INRIA Person Dataset. http://pascal.inrialpes.fr/data/human/
[3] Amr Suleiman, Vivienne Sze, "An Energy-ecient Hardware Implementation of HOG-based Object Detection at 1080HD 60 fps with Multi-scale Support"
[4] Carlo Tomasi, "Histograms of Oriented Gradients"