ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

CUỘC THI SINH VIÊN NGHIÊN CỨU KHOA HỌC

2016 - 2017

NHẬN DẠNG NGƯỜI BẰNG BIỂU ĐỒ CỦA GRADIENT THEO ĐỊNH HƯỚNG

"HUMAN DETECTION BY HISTOGRAM OF ORIENTED GRADIENTS"

Hồ Huy Hùng QH-2014-I/CQ-ĐB

Người hướng dẫn: PGS. TS. Trần Xuân Tú

Hà nội, 2017

Tóm tắt

Nhận dạng đối tượng, đặc biệt là nhận dạng người trong xử lý hình ảnh là một mục tiêu quan trọng và nhiều tiềm năng nhất hiện nay, nó có thể áp dụng vào nhiều ứng dụng thực tiễn như hệ thống hỗ trợ lái xe nâng cao (ADAS) hay điều khiển tự động trong máy bay không người lái (UAV). Một trong những thuật toán phổ biến và thành công nhất trong việc nhận dạng đối tượng là Biểu đồ các Gradient theo định hướng (HOG – Histogram of Oriented Gradients) và Máy vec-tơ hỗ trợ (SVM – Support Vector Machine). Mục tiêu đề tài là tìm giải pháp tối ưu cho nhận dạng người và tiến tới thực thi trên phần cứng. Đề tài trình thuật toán **Nhận dạng người bằng biểu đồ của gradient theo định hướng** (Human Detection by Histogram of Oriented Gradients) để nhận dạng người đi bộ trong ảnh hoặc video. Đề tài được bắt đầu từ cuộc thi LSI Design Contest 2017 Okinawa và từ đó được phát triển rộng hơn. Phương pháp thực hiện trong đề tài là hiểu rõ thuật toán, tìm những nút thắt và giải quyết nó, sau đó mô phỏng kết quả trên phần mềm và cuối cùng là thực thi phần cứng. Môi trường mô phỏng để đánh giá kết quả là ngôn ngữ lập trình C và ngôn ngữ lập trình phần cứng VHDL. Hiện tại đề tài đã đề ra giải pháp giải quyết được nút thắt lớn nhất trong tính toán HOG là sử dụng LUT (Look Up Table) và đang tiến hành mô phỏng thuật toán trên phần cứng.

Từ khóa: Nhận dạng người, Human detection, HOG, SVM

Mục lục

1 Giới thiệu

Ngày nay, máy tính đã trở nên phổ biến trong cuộc sống của chúng ta. Nó có thể thực hiện được những tác vụ có độ phức tạp cao với tốc độ xử lý đáp ứng yêu cầu thời gian thực. Máy tính có thể thay thế con người trong nhiều lĩnh vực phức tạp như cánh tay robot có thể giúp con người mang vác hàng trăm tấn, hay những chiếc xe không người lái đến những nơi nguy hiểm như lõi nhà máy điện hạt nhân, sao Hỏa... Tuy nhiên, vẫn có những thứ mà máy tính chưa thể giúp được nhiều cho con người, một trong số đó chính ta việc thay thế đôi mắt con người. Hàng ngày cuộc sống chúng ta lấp đầy hàng ngàn đối tượng khác nhau, từ nhân tạo như nhà cửa, xe hơi, tàu điện cho đến những thứ tự nhiên như cây cỏ, động vật, núi non, biển cả… chúng ta có khả năng nhận biết được tất cả những thứ đó. Ví dụ như khi nhắc đến một chiếc xe thì có rất nhiều loại như xe hơi, xe tải, xe bus… mỗi xe đều có một màu sắc, kiểu dáng, kích thước khác nhau nhưng chúng ta đều nhận dạng được nó là xe ôtô. Nhưng máy tính hiện tại vẫn còn rất xa trong việc phân tích, suy luận như vậy. Vì vậy, một trong những mục tiêu quan trọng và nhiều tiềm năng nhất hiện nay là nghiên cứu thị giác cho máy tính có khả năng xem, phân tích hình ảnh, video và nhận dạng các đối tượng khác nhau. Nếu có khả năng đó, máy tính sẽ sử dụng cho nhiều ứng dụng công nghệ cao như dây chuyền sản xuất tự động, tương tác người và máy tính, robot, xe tự hành, máy bay không người lái… Biểu đồ của gradient theo định hướng (HOG) kết hợp với Máy vec-tơ hỗ trợ (SVM) là một trong những thuật toán phổ biến và thành công nhất hiện nay trong việc nhận dạng đối tượng, đặc biệt là nhận dạng người trong hình ảnh và video. Vì vậy, trong đề tài này sử dụng HOG-SVM, đặc biệt là tôi đã tìm được điểm tối ưu hóa trong tính toán HOG giúp cải thiện tốc độ nhận dạng người. Đó là sử dụng LUT (Look Up Table) lưu trữ các giá trị thường gặp trong tính toán HOG để chiết xuất nhanh chóng.

Tuy nhiên, việc áp dụng thuật toán xử lý hình ảnh vào các môi trường thực tế lại không đơn giản. Nếu chúng ta thực thi thuật toán trên các hệ thống nhúng phần mềm thì sẽ có rất nhiều khó khăn. Một số lượng lớn pixel ảnh cần xử lý nhiều lần bởi hệ thống để trích xuất những đặc trưng có ích và các đặc trưng này cần kết hợp, xử lý, truy xuất lẫn nhau liên tục. Hệ thống phải xử lý tất các các yêu cầu trên với tốc độ cao và bộ nhớ nhanh để áp dụng những ứng dụng đòi hỏi thời gian thực. Đó là một bài toán nan giải cho các hệ thống nhúng thông thường. Ngược lại, việc thiết kế một phần cứng từ chip trắng

thì sẽ không bị giới hạn về lượng xử lý tức thời nếu có đủ tài nguyên. Khi thiết kế phần cứng, ta có thể xử lý song song các tính toán, điều đó dẫn đến việc giảm đáng kể nhu cầu lưu trữ dữ liệu và sử dụng năng lượng do làm việc với tần số thấp. Hơn nữa, với tần số thấp ta có thể đáp ứng thời gian thực cho các ứng dụng quan trọng như Hệ thống hộ trợ lái xe nâng cao (ADAS) hay điều khiển tự động trong máy bay không người lái (UAV). Trong đề tài này sẽ giới thiệu thuật toán HOG-SVM, các giải pháp tối ưu và thực thi thuật toán lên phần cứng.

2 Nhận dạng người bằng HOG-SVM

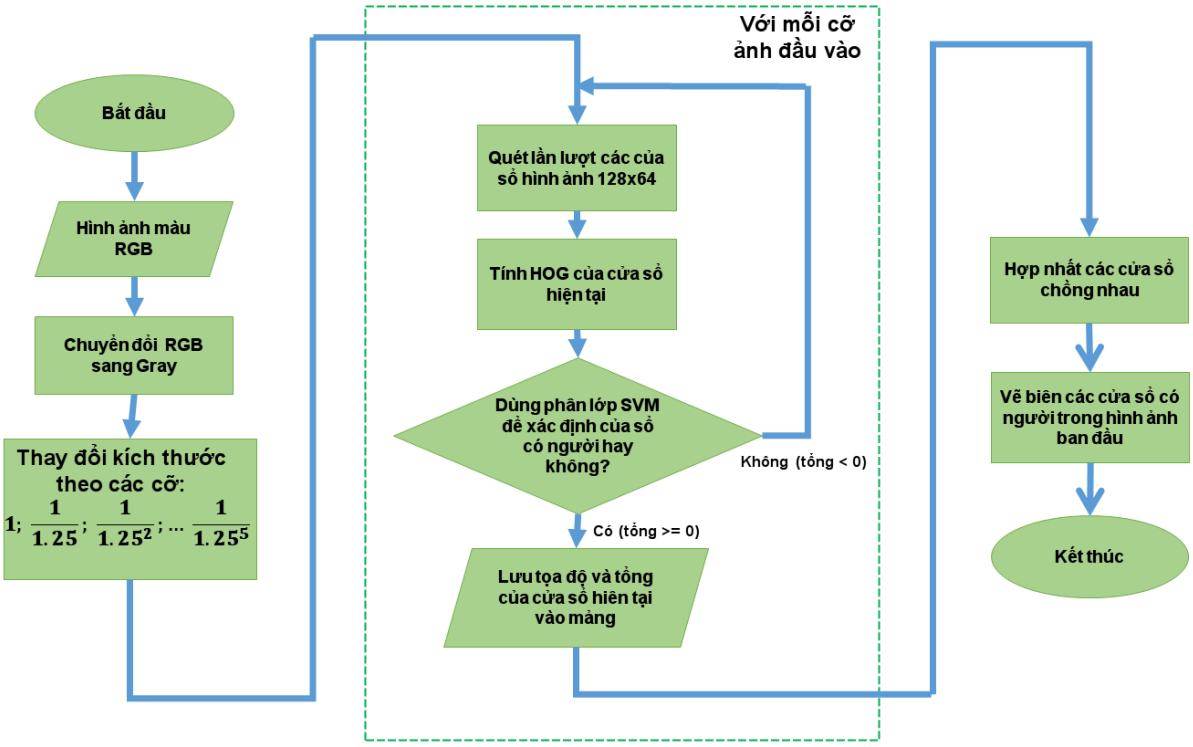
Công việc nhận dạng/xác định đối tượng bao gồm hai quá trình quan trọng: phân tích/tách các đặc trưng trong một khối/vùng ảnh và đánh giá sự trùng khớp giữa những đặc trưng trong khối/vùng ảnh và những hiểu biết hiện tại về vật thể. Tại thời điểm này, HOG là thuật toán thích hợp nhất để chiết xuất đặc trưng trong từng khối/vùng ảnh và bộ nhận dạng SVM đạt được hiệu suất cao nhất với thuật toán HOG [6]. Đặc trưng HOG được hai nhà nghiên cứu N. Dalal và B. Triggs khám phá ra đầu tiên và được mô tả chi tiết trong bài báo của họ [1], từ đó đã có rất nhiều ứng dụng được tạo ra trên cơ sở này.

2.1 Đặc trưng HOG

Biểu đồ của các gradient theo định hướng (HOG – Histogram of Oriented Gradients) là một loại miêu tả đặc trưng của hình ảnh, nó sử dụng kỹ thuật tính số lần xuất hiện các vec-tơ gradient theo các hướng trong khối/vùng ảnh. Khi đó, thuật toán HOG sẽ tính ra được các biểu đồ độ dốc theo các hướng hình ảnh, tạo nên bộ đặc điểm đặc trưng cho đối tượng đang quan tâm. Một bộ miêu tả đặc trưng khác với kỹ thuật tương tự là **Biến đổi đặc trưng bất biến tỷ lệ** (SIFT - Scale-invariant feature transform). Tuy nhiên SIFT mô tả các vùng bất biến đối với các phép dịch, quay, phóng đại… thành các đặc trưng và kết hợp các đặc trưng của mỗi vùng với nhau, vì vậy nó phù hợp cho mục đích sắp xếp, kết hợp hình ảnh hay nhận diện đối tượng cho trước. Còn HOG chiết xuất đặc trưng từ các "ô" dày đặc chồng lên nhau trong từng khối/vùng ảnh và kiểm tra sự tương đồng giữa đặc trưng này với đặc trưng của đối tượng cần tìm. Thuật toán HOG

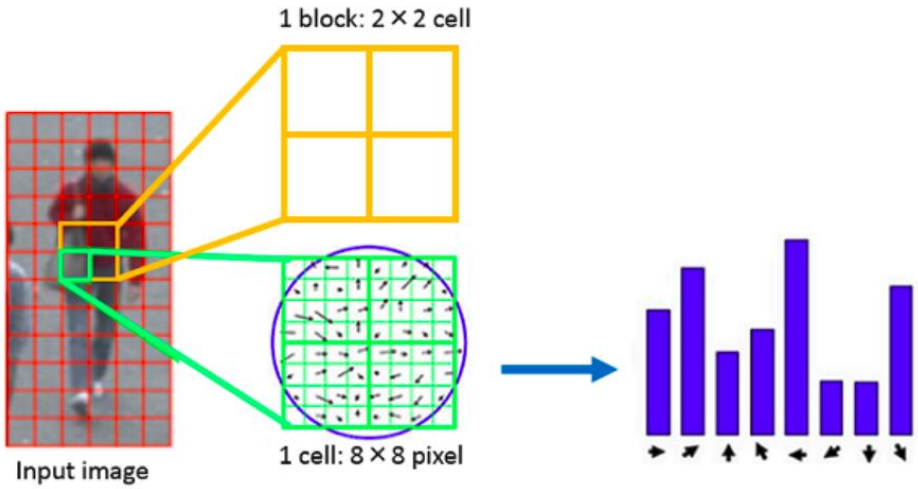
sẽ quét lần lượt các cửa sổ trên toàn bộ hình ảnh để xác nhận có những đối tượng nào cần tìm trong hình ảnh và video.

Trình tự các bước nhận dạng người trong đề tài này được tóm tắt trong sơ đồ sau (các tham số có thể thay đổi tùy vào mục đích sử dụng khác nhau):



Hình 1: Thuật toán HOG-SVM.

Với mỗi cỡ ảnh đầu vào, ta lần lượt quét các cửa sổ hình ảnh 128x64 pixel phù hợp cho việc nhận dạng người đi bộ [1]. Chia cửa sổ hình ảnh thành các khối 8x8 (cell) vàcác khối 16x16 (block). Trình tự để chiết xuất đặc tính HOG được tóm tắt như sau:



Hình 2: Tính toán biểu đồ.

(1) Tính xấp xỉ độ lớn vec-tơ gradient (là vec-tơ có các thành phần biểu thị tốc độ thay đổi giá trị của điểm ảnh) của từng pixel ảnh sử dụng bộ lọc [1 0 1] và $[1\ 0\ 1]^T$:

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

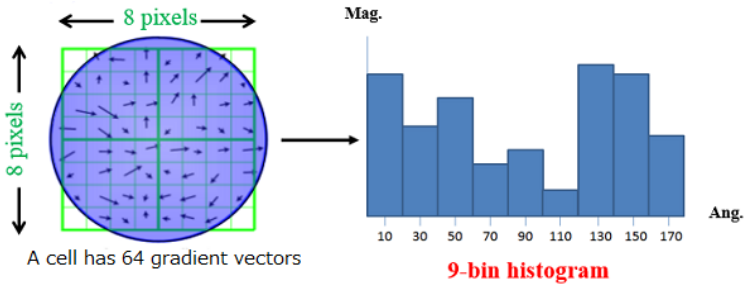
$$f_y(x,y) = I(x,y+1) - I(x,y-1)$$

(2) Tính biên độ và hướng của vec-tơ gradient trong hệ tọa độ cực:

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

$$\theta(x,y) = \arctan(\frac{f_y(x,y)}{f_x(x,y)})$$

(3) Mỗi khối 8x8 sẽ được thể hiện bằng một biểu đồ gradient gồm 9 cột (bin) với các hướng khác nhau. Mỗi một pixel sẽ bỏ phiếu tỉ lệ thuận với các cột có hướng gần với hướng vec-tơ gradient của pixel ấy nhất.



Hình 3: Bỏ phiếu biểu đồ của mỗi khối 8x8.

(4) Giá trị của 9 cột trong biểu đồ dựa trên độ lớn gradient trong các pixel của khối 8x8 mà đã được tính. Giả sử nếu mỗi pixel có độ lớn tăng lên 2 lần, thì độ lớn của các cột trong biểu đồ cũng tăng 2 lần, dẫn đến đặc trưng HOG cũng bị thay đổi. Bằng cách chuẩn hóa biểu đồ, ta có thể khiến nó bất biến đối với sự thay đổi độ sáng (hay là độ tương phản). Ta nhóm 4 khối 8x8 gần nhau thành khối 16x16 để chuẩn hóa, hợp thành một vec-tơ 36 phần tử. Chia vec-tơ này bằng độ lớn trung bình để chuẩn hóa:

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

$v(k)$ là giá trị thứ k của vec-tơ HOG, q là kích thước khối 8x8, N là số khối 8x8 trong 1 khối 16x16.

ε là hằng số nhỏ để loại bỏ phép chia với 0.

(5) Vec-tơ đã chuẩn hóa của các khối 16x16 kết hợp với nhau tạo thành đặc trưng HOG. Trong đề tài này sử dụng đơn vị khối cỡ 8x8 và cửa sổ cỡ 128x64 pixel nên kích thước của đặc trưng HOG là 3780 phần tử. Đây là một trong những nút thắt cổ chai của thuật toán, kết quả sẽ sử dụng nhiều bộ nhớ và tính toán phức tạp.

(6) Đặc trưng HOG được đưa vào bộ phân loại SVM để xác định đối tượng cần quan tâm có mặt trong cửa sổ hiện tại hay không.

Máy vec-tơ hỗ trợ (SVM – Support Vector Machine) là một loại học máy (Machine learning) cho phân loại, trong đề tài này sử dụng SVMs tuyến tính cho phân loại đặc tính HOG. Trước tiên, đào tạo SVM sử dụng nhiều khung hình đã xác nhận có người hay không từ bộ dữ liệu INRIA Person Dataset [2], kết quả được một vec-tơ lưu trữ đặc tính HOG nổi bật của một người đi bộ. Nếu đào tạo vec-tơ HOG bằng SVM từ bộ dữ liệu càng lớn thì độ chích xác khi nhận dạng người càng cao. Sau đó so sánh vec-tơ đã đào tạo này với đặc trưng HOG của cửa sổ này, ta nhận ra có người trong cửa sổ có người hay không.

Bộ phân loại SVM phân loại quyết định nhị phân, được xác định bởi hàm:

$$y(x) = w \,.\, x + b$$

Trong đó, x là đặc trưng HOG, w là vec-tơ SVM đã được đào tạo, b là hằng số lệch.

Hàm quyết định cửa sổ k có đối tượng hay không:

$$f(x_k) = \begin{cases} 1 & y(x_k) > 0 \\ 0 & y(x_k) \leq 0 \end{cases}$$

Thuật toán cơ bản không có khả năng nhận biết kích thước/khoảng cách đối tượng trong hình ảnh, vì vậy cần có một phương án để nhận dạng được đối tượng kích thước khác nhau. Phương án đầu tiên là sử dụng nhiều bộ phân loại SVM, đã được đào tạo cho cùng một đối tượng với các kích cỡ khác nhau. Cách này lộ rõ nhược điểm là tăng sự phức tạp của quá trình đào tạo và tăng bộ nhớ lưu trữ. Vì vậy thuật toán trong đề tài này sử dụng phương án thứ hai: thay đổi kích thước của khung hình, sau đó được xử lý với một máy phân loại SVM duy nhất. Phương án này loại bỏ được các nhược điểm của phương án thứ nhất, tuy nhiên nó cũng có hạn chế là gia tăng tính toán trong bộ thay đổi kích thước ảnh.

Hình 4: Kích cỡ của một người trong hình ảnh phụ thước vào khoảng cách từ camera đến người đó, vì vậy cần thay đổi kích thước hình ảnh với nhiều cỡ khác nhau để quét.

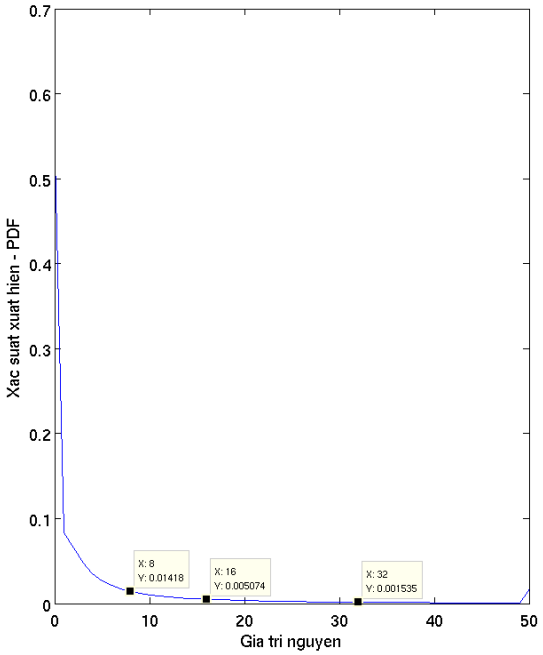
2.2 Nút thắt trong tính toán HOG

Trong thuật toán HOG, độ lớn gradient được tính xấp xỉ bằng độ sai khác giữa các điểm ảnh lân cận:

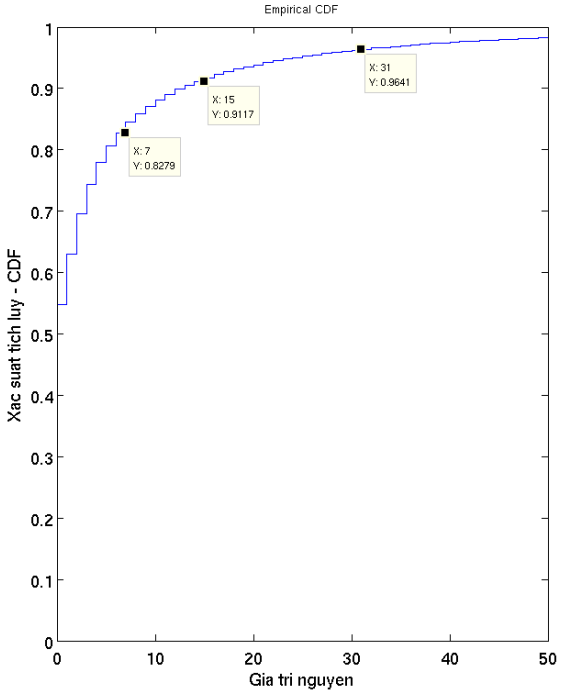
$$f_x(x, y) = I(x + 1, y) - I(x - 1, y)$$

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

Trong một bức ảnh số, các điểm ảnh gần nhau có mối liên hệ nhất định về giá trị. Hình 5 và hình 6 trình bày mật độ xác suất và xác xuất tích lũy về trị tuyệt đối sai khác giữa các điểm ảnh lân cận nhau: (x + 1) và (x - 1); (y + 1) và (y - 1). Xác suất tính từ 10 ảnh ngẫu nhiêu trong bộ dữ liệu chuẩn INRIA Person Dataset [2]:



Hình 5: Xác suất xuất hiện độ lớn gradient, xác suất xuất hiện giá trị 16 chỉ khoảng 0.5%.



Hình 6: Xác suất tích lũy của độ lớn gradient. Xác suất xuất hiện độ lớn nhỏ hơn 16 là hơn 91%.

Khó khăn hàng đầu trong thuật toán HOG chính là khối lượng tính toán rất lớn. Trong HOG, việc tính độ lớn và góc gradient trong hệ toạ độ cực sử dụng các phép tính toán học có độ phức tạp lớn: bình phương, căn bậc 2 và arctan.

$$m(x,y) = \sqrt{f_x(x,y)^2 + f_y(x,y)^2}$$

$$\theta(x,y) = \arctan\left(\frac{f_y(x,y)}{f_x(x,y)}\right)$$

Vì vậy, với với mỗi điểm ảnh, thuật toán HOG yêu cầu số lượng phép tính rất lớn. Đây chính là nút thắt lớn nhất để đáp ứng yêu cầu tính toán HOG trong thời gian thực, đặc biệt là trong các ứng dụng yêu cầu tốc độ xử lý cao như ô-tô tự động, máy bay không người lái.

Đề xuất của đề tài này là sử dụng LUT (Look up Table) trong việc chuyển đổi gradient sang hệ tọa độ cực. Tức là ta sẽ lưu các giá trị độ lớn và góc của các giá trị

thường gặp vào trong LUT (bộ nhớ), sau đó khi cần sử dụng ta chỉ cần đọc dữ liệu từ bộ nhớ ra. Chắc chắn là việc đọc dữ liệu từ bộ nhớ sẽ nhanh hơn nhiều so với các phép toán nhân, căn bậc 2, arctan.

Đặt $a = \mathrm{f_x}(\mathrm{x},\mathrm{y})$, $b = \mathrm{f_y}(\mathrm{x},\mathrm{y})$. Nếu a và b đồng thời nhỏ hơn 16 thì trực tiếp xuất giá trị của LUT từ bộ nhớ ra. Trường hợp a và b không đồng thời nhỏ hơn 16 thì thực hiện phép tính độ lớn và góc của vec-tơ gradient như bình thường.

Bây giờ chúng ta giả sử hàm đọc dữ liệu LUT từ bộ nhớ và các phép tính khác như phép so sánh, phép nhân, phép chia, phép căn bậc hai cùng độ phức tạp là O(1). Khi đó phép tính khi sử dụng LUT được phân chia như sau:

Mã giả	Số phép tính
$if\ (a < 16\ and\ b < 16)$	2
$\mathrm{m}(\mathrm{x},\mathrm{y}) = LUT\_magnit$	1
$\theta(x,y) = LUT\_angle$	1
$else$	
$\mathrm{m}(\mathrm{x},\mathrm{y}) = \sqrt{a^2 + b^2}$	4
$\theta(x,y) = \arctan(\frac{b}{a})$	2

Ta xét thuật toán xảy ra 10 lần và coi xác suất xuất hiện đồng thời a và b nhỏ hơn 16 là P. Tổng số phép tính xảy ra là :

$$C(1) = 2 \times 10 + P + P + (10 - P) \times 4 + (10 - p) \times 2 = 80 - 4 \times P$$

Nếu sử dụng tính toán thông thường, tổng số phép tính khi thực hiện 10 lần là :

$$C(2) = 10 \times 4 + 10 \times 2 = 60$$

Để tổng số phép tính trong thuật toán LUT ít hơn tổng số phép tính bình thường thì :

$$C(1) < C(2) \leftrightarrow 80 - 4 \times P < 60 \leftrightarrow P > 5$$

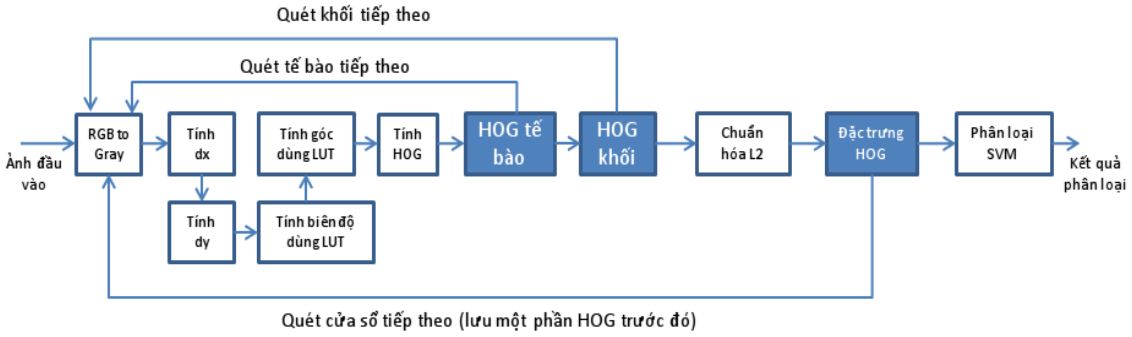
Điều này có nghĩa là với xác suất xuất hiện của a và b nhỏ hơn 16 là lớn hơn 50% thì số phép toán khi sử dụng LUT sẽ nhỏ hơn. Trong khi xác suất thực nghiệm ở hình 6 cho thấy xác suất này hơn 91%. Vì vậy sử dụng LUT là một sự tối ưu cực kì hiệu quả để giảm thiểu khối lượng thực thi thuật toán.

Một thách thức thứ hai của thuật toán HOG, đó chính là độ dài 3780 của đặc trưng HOG cuối cùng. Kết quả này sẽ phải sử dụng nhiều bộ nhớ và tính toán phân loại SVM phức tạp. Đề tài này sử dụng tính toán song song trong phần cứng sẽ cân bằng được thời gian tính toán và hiệu năng, chi tiết điều này sẽ được trình bày rõ hơn ở phần sau.

3 Mô phỏng phần mềm trên C

3.1 Kiến trúc

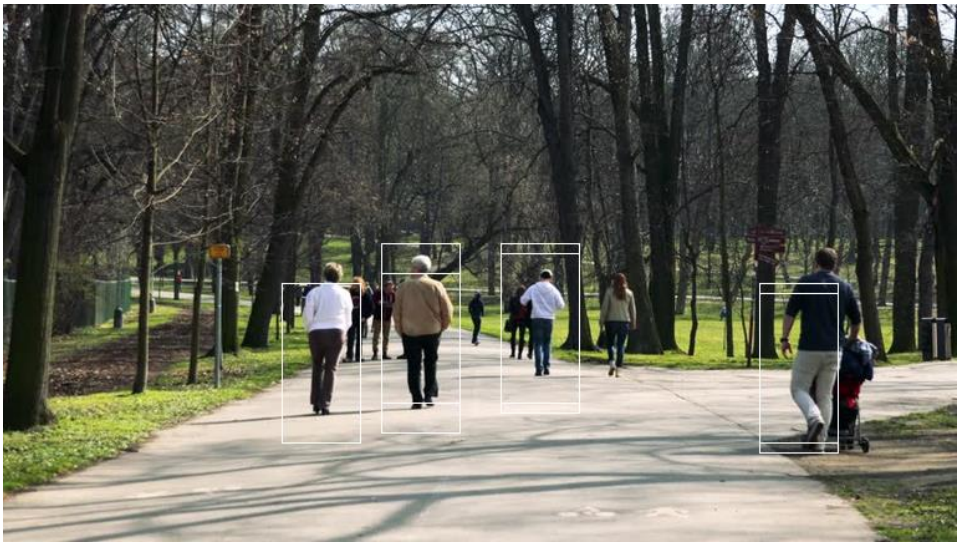
Sơ đồ kiến trúc trên C được thể hiện như hình 7. Việc thay đổi kích cỡ ảnh được thực hiện gián tiếp qua lệnh convert trên shell linux.



Hình 7: Kiến trúc HOG-SVM trên C.

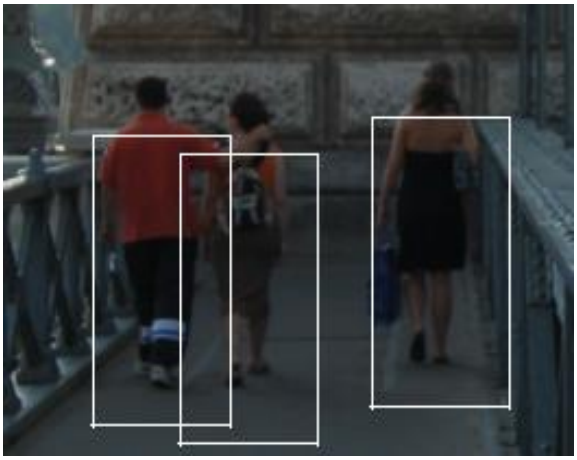
3.2 Kết quả mô phỏng

Mô phỏng bằng ngôn ngữ lập trình C được thực hiện sử dụng gcc, hệ điều hành Ubuntu 16.04.



Hình 8: Nhận diện được 4/5 người trong một cỡ.

Hình 9: Nhận diện người khó nhìn thấy.



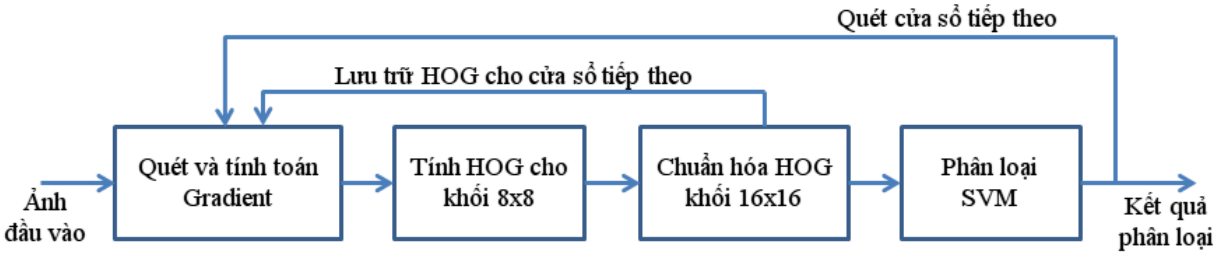
Hình 10: Nhận diện 3/3 người.

4 Kiến trúc phần cứng

Với kết quả đã mô phỏng trên C, ta hoàn toàn có thể nhúng thuật toán trên một con vi xử lý mạnh mẽ để ứng dụng. Tuy nhiên, giới hạn của phần mềm không đủ để có thể xử lý thời gian thực với HOG-SVM. Dù thời gian thực có đảm bảo, các ràng buộc về tốc độ cao hay năng lượng thấp cũng không hề dễ dàng thực hiện được. Qua quá trình mô phỏng phần mềm và tài liệu [5] kết luận rằng thuật toán HOG-SVM trên phần mềm không thể đạt được thông lượng quá 10 FPS. Do vậy giải pháp phần mềm hoàn toàn không phù hợp. Trong phần này đề xuất việc thực thi phần cứng của bộ nhận dạng đối tượng HOG-SVM dựa trên cơ sở tài liệu [3], [4] cho hệ thống đủ khả năng ứng dụng thời gian thực.

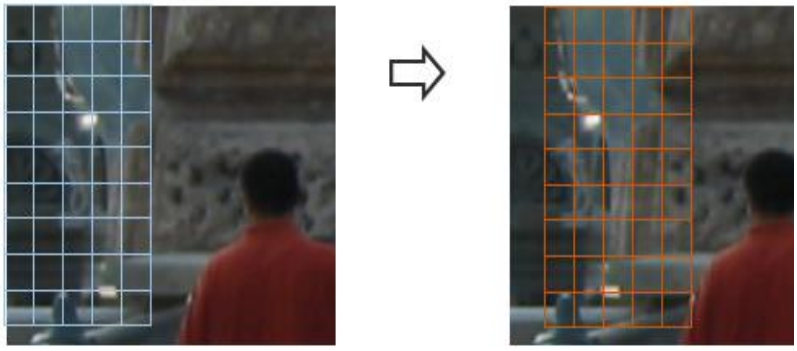
Hình 11 thể hiện kiến trúc phần cứng thuật toán HOG-SVM với 4 khối chính:

- Khối quét ảnh và tính toán gradient: Tính toán sử dụng LUT để tối ưu hóa (phần 4.1), tính toán góc gradient có thể dùng thuật toán CORDIC để tăng tốc
- Khối HOG của khối 8x8: Xấp xỉ bỏ phiếu cho các cột trong biểu đồ mỗi khối
- Khối chuẩn hóa HOG khối 16x16: sử dụng chuẩn hóa L2 (phần 4.2)
- Khối phân lớp SVM: Tính toán SVM song song (phần 4.3)

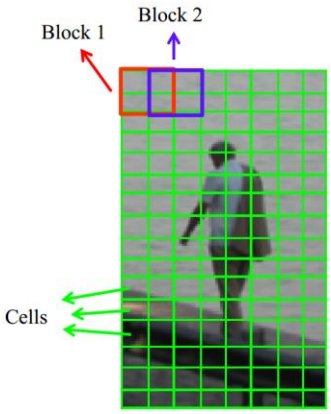


Hình 11: Kiến trúc phần cứng thuật toán HOG-SVM.

4.1 Khối quét ảnh và tính toán gradient



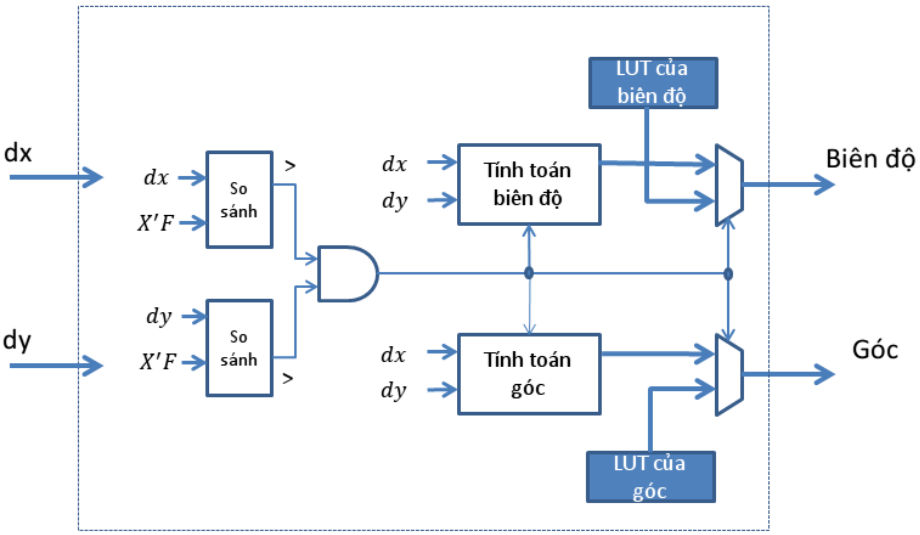
Hình 12: Quét cửa sổ hình ảnh tiếp theo.



Hình 13: Quét khối tiếp theo.

Xác nhận đối tượng với tính năng HOG được thực hiện bởi cửa sổ dò tìm và kích thước cửa sổ này phải phù hợp đối với đối tượng cần nhận dạng, kích thước tốt nhất để nhận dạng người đi bộ là 128x64 pixel. Sau khi một cửa sổ được tính đặc trưng HOG và nhận dạng người dùng SVM, cửa sổ tiếp theo sẽ được quét sử dụng một phần đặc trưng HOG của cửa sổ trước đó như hình 12. Hình 13 cho thấy cách quét các khối 8x8 từ trái qua phải. Đặc trưng HOG được tính từ các phần tử nhỏ nhất là các khối 8x8, không có khối nào chồng lên khối khác, chỉ có các khối 16x16 chồng lên nhau. Chia sẻ biểu đồ HOG của khối 8x8 và tái sử dụng trong khối 16x16 tiếp theo và cửa sổ tiếp theo là cần thiết, có tác động lớn đến việc giảm băng thông bộ nhớ.

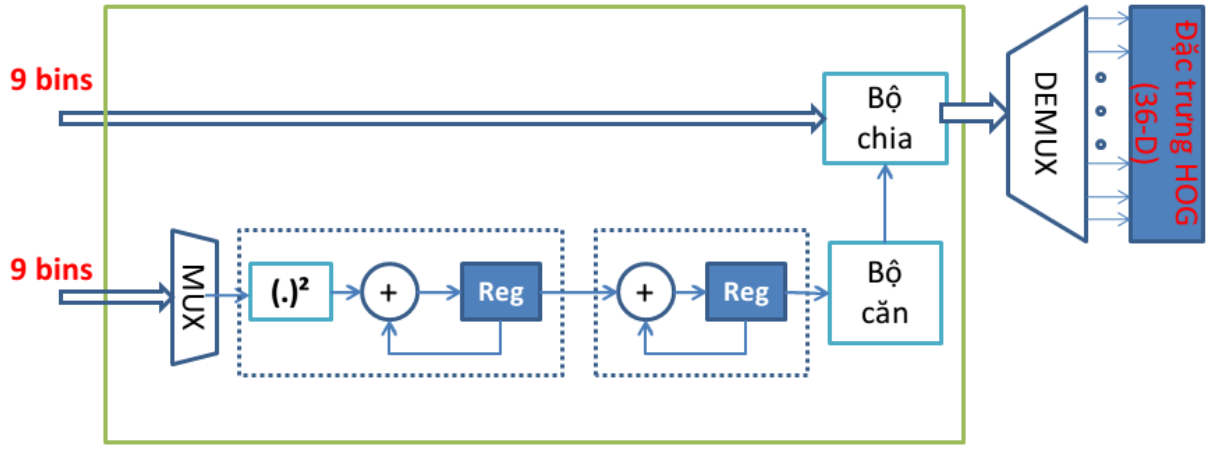
Tính toán Gradient sử dụng LUT (Look Up Table) đã đề xuất ở trên được biểu diễn như hình 14 phía dưới: sử dụng 2 bộ so sánh dx, dy với giá trị 16, sau đó kết quả so sánh được dùng làm tín hiệu điều khiển cho bộ tính toán biên độ và bộ tính toán góc:



Hình 14: Sử dụng LUT tính biên độ và góc Gradient.

4.2 Khối chuẩn hóa biểu đồ

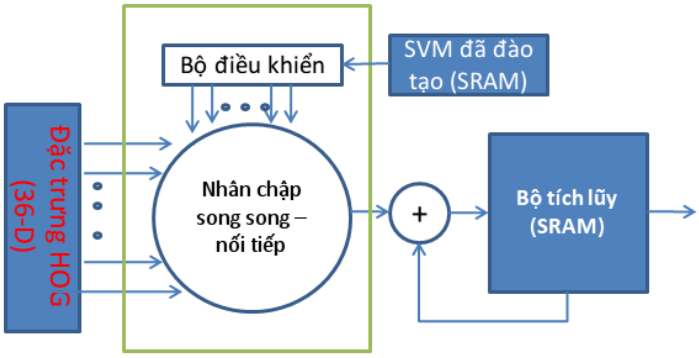
Như đã nhắc đến ở phần trước, mỗi khối 8x8 cần 8 khối 8x8 ở lân cận để tạo ra 4 khối thực hiện chuẩn hóa chồng nhau. Vì vậy đặc trưng HOG của mỗi khối 8x8 phải được lưu trong bộ đệm để sử dụng tính toán chuẩn hóa đối với các khối khác nhau. Tuy nhiên vẫn còn nút thắt trong khối này khi cần sử dụng bộ căn bậc 2 và bộ chia như hình 15 mô tả:



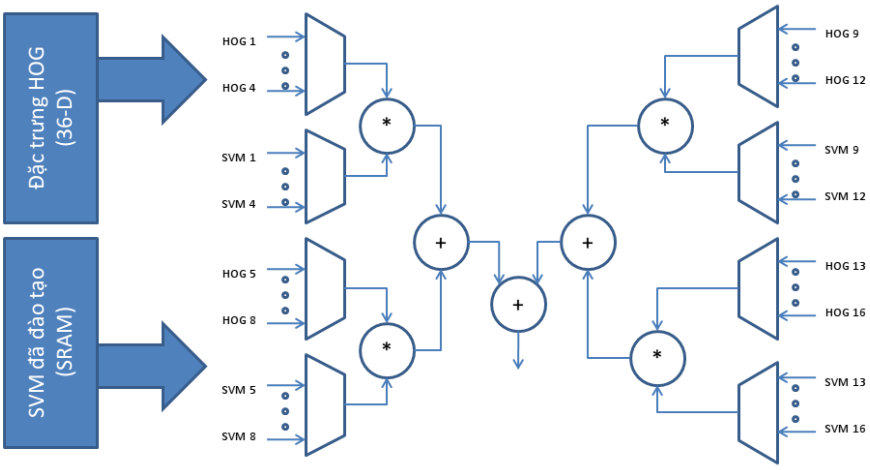
Hình 15: Khối chuẩn hóa biểu đồ.

4.3 Khối phân lớp SVM

Một bộ phân loại SVM tuyến tính đã được đào tạo sẽ lưu trong SRAM, sau khi chiết xuất được đặc trưng HOG, ta tiến hành tính toán SVM. Hình 16 cho thấy đặc trưng HOG của mỗi khối 8x8 được sử dụng ngay lập tức để phân loại, vì vậy sẽ không bao giờ phải tính lại trong các cửa sổ khác. Tính toán SVM được mô tả như hình 16 và hình 17, sử dụng song song kết hợp nối tiếp khi nhân chập với 4 bộ nhân để đảm bảo cân bằng giữa hiệu suất và yêu cầu phần cứng:



Hình 16: Sơ đồ khối phân lớp SVM.



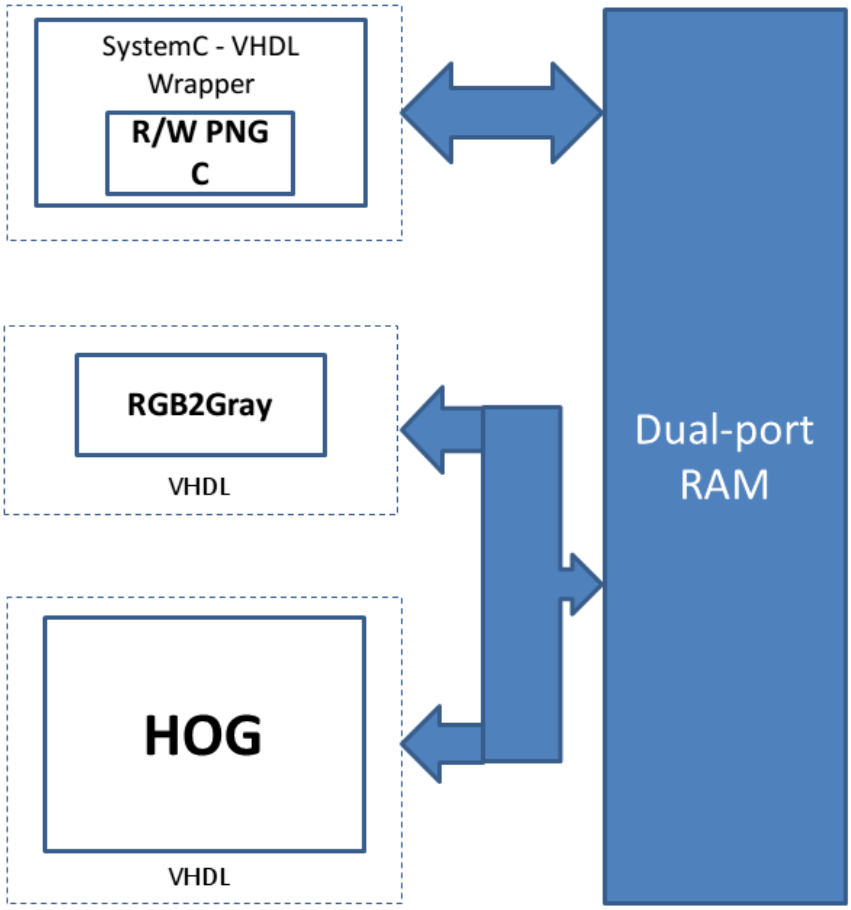
Hình 17: Tính toán SVM bằng nhân chập song song kết hợp nối tiếp.

5 Mô phỏng phần cứng trên VHDL

Đề tài này sử dụng ngôn ngữ mô tả phần cứng VHDL, mức mô tả chuyển dịch thanh ghi (RTL) để mô phỏng kiến trúc đề xuất.

5.1 Môi trường kiểm thử

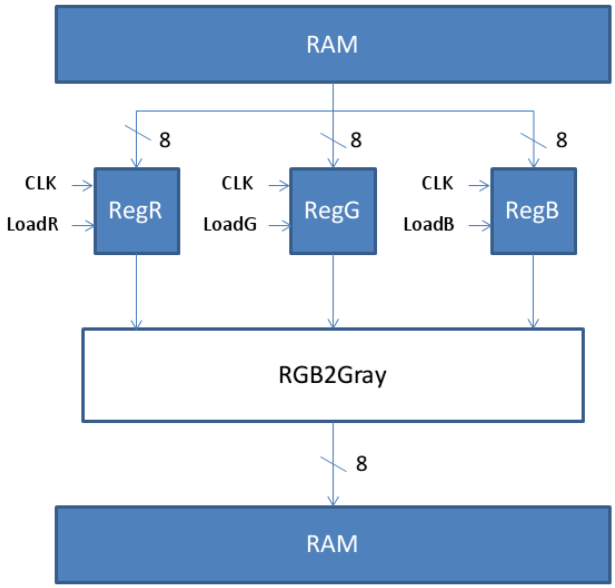
Việc mô phỏng sẽ kết hợp phần cứng và phần mềm, đọc và lưu ảnh sẽ sử dụng mã nguồn ngôn ngữ C, thông qua giao tiếp SystemC-VHDL là lưu trong Dual-port RAM, còn khối RGB2Gray và khối HOG viết bằng VHDL sẽ giao tiếp với Dual-port RAM.



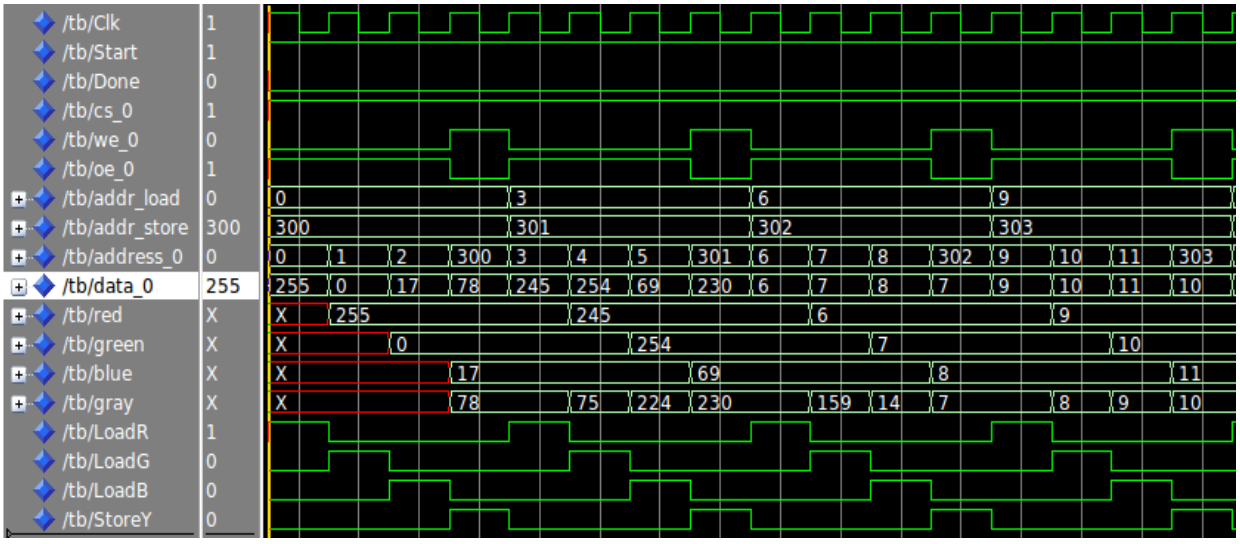
Hình 18: Giao diện mô phỏng.

5.2 Bộ chuyển ảnh RGB sang Gray

Hình 18 thể hiện kiến trúc chi tiết của bộ chuyển ảnh RGB sang Gray và hình 19 là kết quả mô phỏng. Mỗi pixel được chuyển sang Gray với 4 chu kì đồng hồ: 3 chu kì đầu lấy dữ liệu lần lượt từng thành phần RGB từ RAM, chu kì thứ 4 tính toán giá trị Gray và lưu vào RAM ở vị trí lưu dành cho ảnh Gray.



Hình 19: Kiến trúc bộ chuyển RGB sang Gray.



Hình 20: Kết quả mô phỏng bộ chuyển RGB sang Gray.

6 Kết luận

Đề tài đã đề xuất một điểm tối ưu hóa quan trọng giúp tăng tốc thực thi thuật toán trong chủ đề nhận dạng người bằng biểu đồ các Gradient định hướng. Tuy nhiên cần có thêm thời gian đề có thể mô phỏng kết quả trên mạch cứng.

Trong tương lại, đề tài sẽ hoàn thành mô phỏng phần cứng thuật toán bằng VHDL, đặc biệt là mô phỏng việc tối ưu hóa sử dụng LUT. Mô phỏng mức chuyển dịch thanh ghi (RTL) có khả năng thực thi trên FPGA và đưa vào ứng dụng.

Tài liệu tham khảo

[1] Dalal, N. & Triggs, B. (2005), "Histograms of oriented gradients for human detection". In Proceedings IEEE Computer Society Conference on Computer Vision and PatternRecognition

[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] Kosuke Mizuno, (2012), "Architectural Study of HOG Feature Extraction Processor for Real-Time Object Detection". In IEEE Workshop on Signal Processing Systems.

[5] Hakim, V.S. El (2015), "Implementation and Analysis of Real-time Object Tracking on the Starburst MPSoC"

[6] H. Bristow and S. Lucey, "Why do linear svms trained on HOG features perform so well?," CoRR, vol. abs/1406.2419, 2014

[7] Kosuke Mizuno, "Implementation and Analysis of Real-time Object Tracking on the Starburst MPSoC", ieice trans. electron., vol.e96-c, no.4 april 2013