Chi tiết thuật toán đã làm:

Extraction HOG histogram:

1. Normalize gamma, dung hàm “equalizeHist” trong OpenCv cho image.
2. Tính gradient tại từng pixel.

( Em hiểu gradient ở đây là 1 vector, gồm có chiều và độ lớn.)

Em dùng mask là [-1 0 1] cho Ox và [-1 0 1]’ cho Oy. Tại mỗi kênh màu của image, em dung hàm imFilter trong OpenCV để tính Dx, Dy. Rồi dung Dx, Dy này để tính gradient tại từng pixel. Cách tính như sau:

Weight =

Orientation = arctan(Dy/Dx);

Tại mỗi pixel, ta sẽ lựa chọn kênh màu nào có độ dài của gradient lớn nhất (ko quan tâm tới hướng của gradient).

1. (Optional) Smooth block (*lúc này chưa tính histogram*)

Hiện giờ ta có 2 matrix 2 chiều: matrix của độ lớn và matrix của hướng trong gradient. Ta sẽ smooth block cho từng matrix.

+ Xác định độ lớn của block trong matrix (*chiều của matrix bằng với chiều của image*), độ lớn của block là (kích thước block)\*(kích thước cell), sau đó dung hàm GaussianBlur với tham số kSize(3,3), sigma = 0.5\*(chiều dài cell)\*(chiều dài block).

+Làm lần lượt như vậy cho các block còn lại.

Pixel

Orientation

Weight

Smooth block for weight and orientation separately

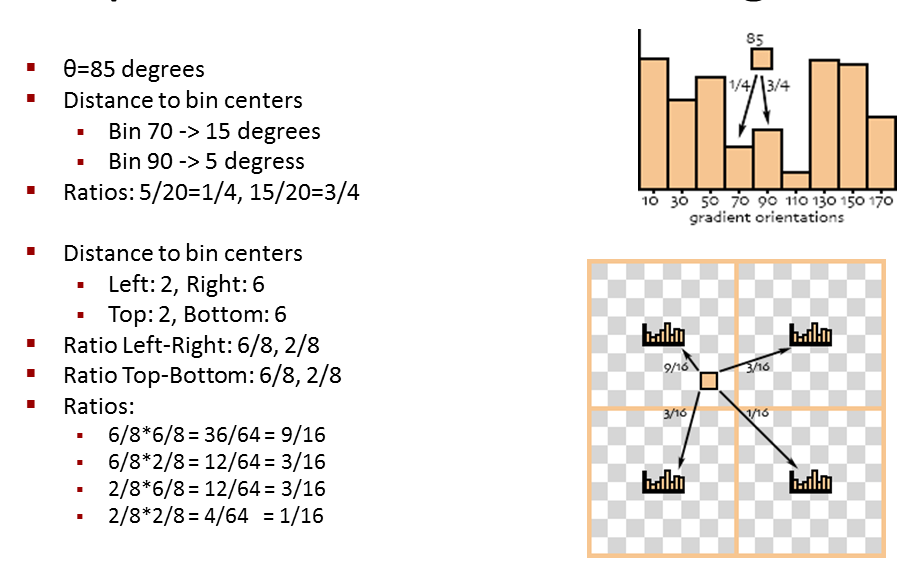
Cell

Block

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

1. Tính histogram của cell

Tại mỗi pixel, ta sẽ xác định gradient của nó đóng góp bao nhiêu % cho cell chứa nó và các cell lân cận . Minh hoa boi hinh sau:



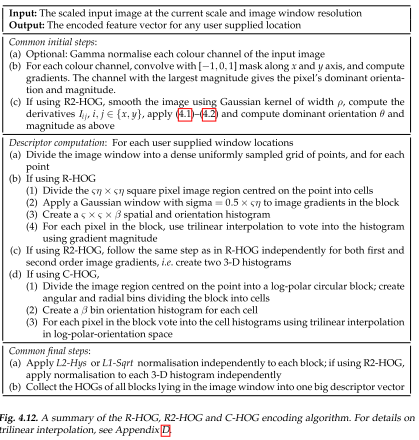
1. Tính histogram của block và normalize block

Ghép các histogram của các cell trong block lại theo thứ tự từ trái sang phải, từ trên xuống dưới. Sau đó normalize block, dung L1 sqrt.

1. Tính histoagram của window

Ghép các histogram của các overlap block theo thứ tự từ trái sang phải, từ trên xuống dưới.

Thật toán mà Dalal dung là như sau:



Cách làm thí nghiệm

1. Dataset: INRIA pedestrian dataset 2009,
2. Training phase

* Positive: 2416 windows, negative: 1218 images. Trong image thì ta có thể dùng sliding window đ ể lấy ra nhiều windows.
* Cách học:
* Tạo ra file train.txt gồm 2416 hàng positive và khoảng 14000 hàng negative.
* Tạo các file val\_xx.txt, c ác file này chỉ chứa phần tử negative, m ục đích là để học hard negative sample. Mỗi file có khoảng 28000 phần tử.
* Đưa các tập val\_xx.txt vào để test và tìm ra những phần tử False-Positive, sau đó đưa các phần tử False-Positive này vào tập train.txt rồi train lại.

1. Testing phase

* Positive: 1126 windows, negative: 453 images.
* Kết quả như sau:

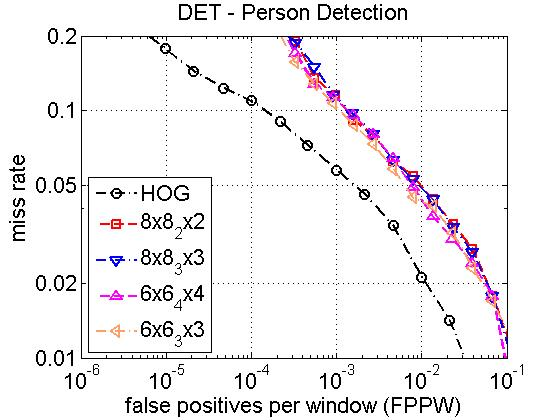


Figure 1: Performance of Dalal and us. Dalal’s curve is the first one (black).