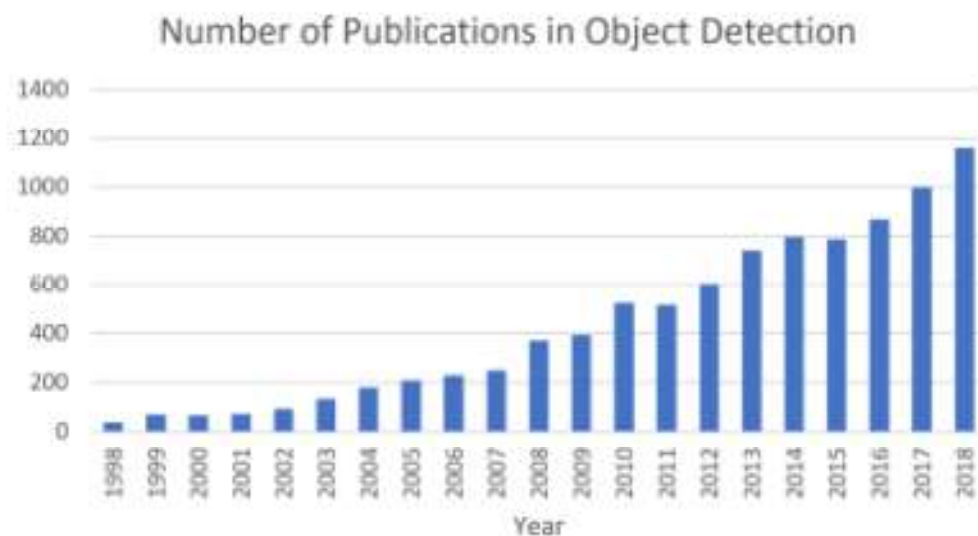


Object Detection Traditional Detector

Viola Jones & HOG Algorithm

2021210088 허지혜

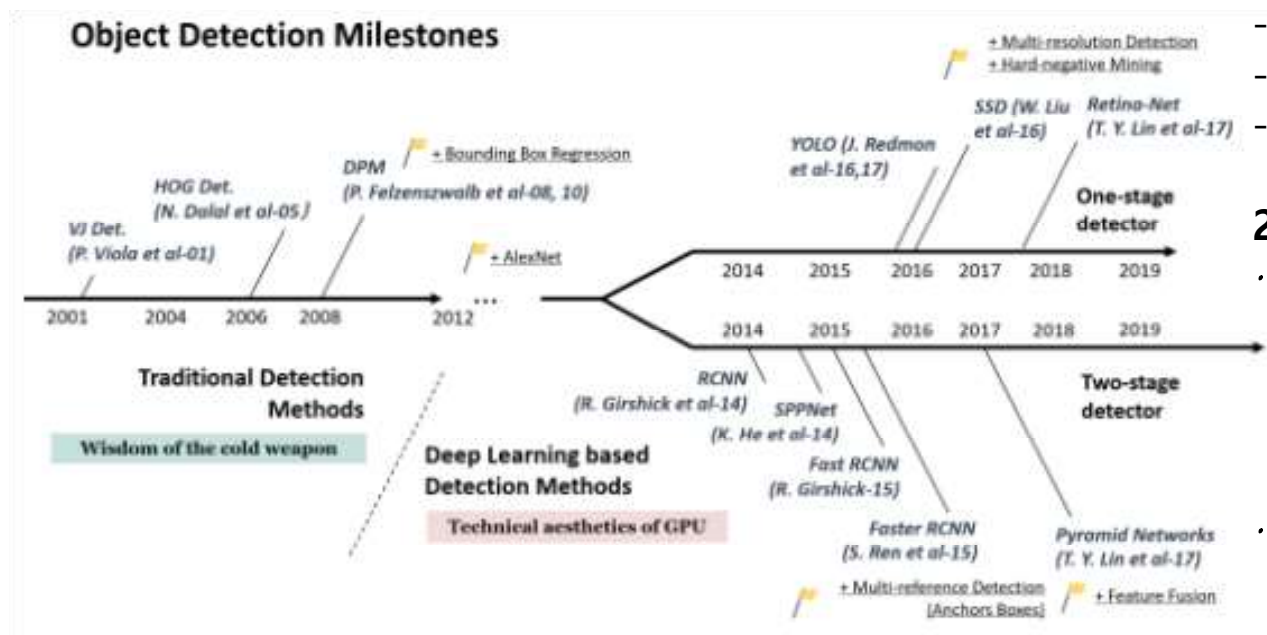
Object Detection in 20 Years : A survey



- Object Detection 논문의 수가 증가하고 있다.
- > 그만큼 중요하다.

Fig. 1. The increasing number of publications in object detection from 1998 to 2018. (Data from Google scholar advanced search: *allintitle: "object detection" AND "detecting objects"*.)

Object Detection in 20 Years : A survey



1. Traditional Detection Methods

- Viola Jones Algorithm
- HOG Algorithm
- DPM

2. Deep Learning based Detection Methods

· One-stage Detector

- YOLO
- SSD
- Retina-Net

· Two-stage Detector

- R-CNN
- SPPNet
- Pyramid Networks

Viola Jones

- Abstract

- OD를 위한 머신러닝 접근 방식을 설명하는 논문
- Face Detection에서 빠르고 높은 탐지율
- 새로운 이미지 표현 도입 (Integral Image)
- AdaBoost 기반 알고리즘으로 효율적인 분류기
- background를 허용하는 Cadscale 사용
- Real-time에서 초당 15 Frame마다 Detector

Viola Jones

Integral Image

Haar Feature

AdaBoost Algorithm

Cascade 구조

Viola Jones

Input
Image



Haar
Feature



Integral
Image



AdaBoost
Training



Cascading
Classifiers

Viola Jones

- Haar Feature

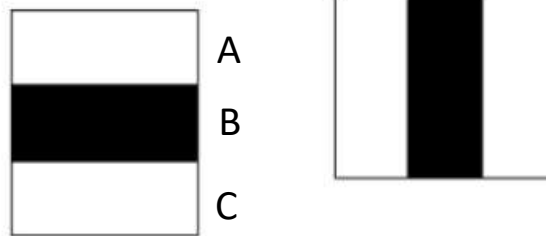
Edge Feature



Value
(하얀색 픽셀 합)-(검정색 픽셀 합)

6개의 점이 나옴

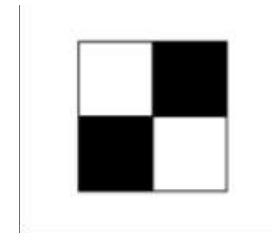
Line Feature



Value
(하얀색 픽셀 합)-(검정색 픽셀 합)
 $= (A+C)-B$

8개의 점이 나옴

Four-rectangle Feature



Value
(하얀색 픽셀 합)-(검정색 픽셀 합)
 $= \text{대각선 쌍 차이}$

9개의 점이 나옴

사람 얼굴의
명암을 특징으로 사용

Viola Jones



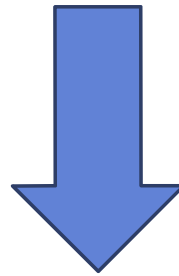
첫번째 사진. 원본

두번째 사진. Edge Feature를 이용하여 눈임을 인지
-> 눈이 그늘지기 때문에 픽셀 사이에 차이가 있을 것

Figure 3: The first and second features selected by Adaboost. The two features are shown in the top row and then overlayed on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

세번째 사진. Line Feature를 이용하여 코임을 인지
-> 코 등이 밝기 때문에 픽셀 사이에 차이가 있을 것

24x24 window에서 180,000개
이상의 feature가 나옴



연산량이 많아져 시간 오래 걸리는 단점

Viola Jones

- Integral Image

- 적분 이미지? 통합 이미지?
- 일반 이미지에서 픽셀값을 이동시킬 때 가중합을 적어주는 것
- 일반 이미지보다 빠르게 계산을 할 수 있다.

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y'),$$

적분 이미지 방정식

Viola Jones

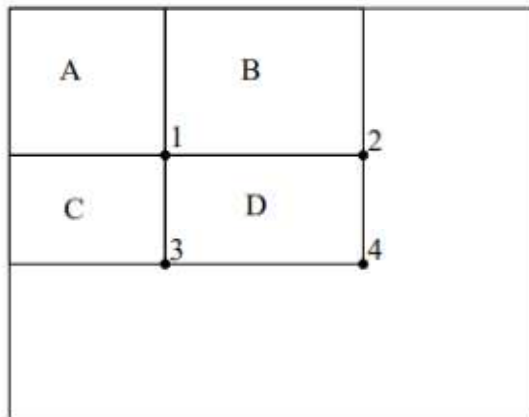


Figure 2: The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A . The value at location 2 is $A + B$, at location 3 is $A + C$, and at location 4 is $A + B + C + D$. The sum within D can be computed as $4 + 1 - (2 + 3)$.

1 : A

2 : $A+B$

3 : $A+C$

4 : $A+B+C+D$

D의 값을 구하기 위해서 $4+1-(2+3)$

$4+1-(2+3)$

$(A+B+C+D+A)-(A+B+A+C) = D$

Viola Jones

64	2	3	61	60	6	7	57
9	55	54	12	13	51	50	16
17	47	46	20	21	43	43	24
40	26	27	37	36	30	31	33
32	34	35	29	28	38	39	25
41	23	22	44	45	19	18	48
49	15	14	52	53	11	10	56
8	58	59	5	4	62	63	1

Original Image

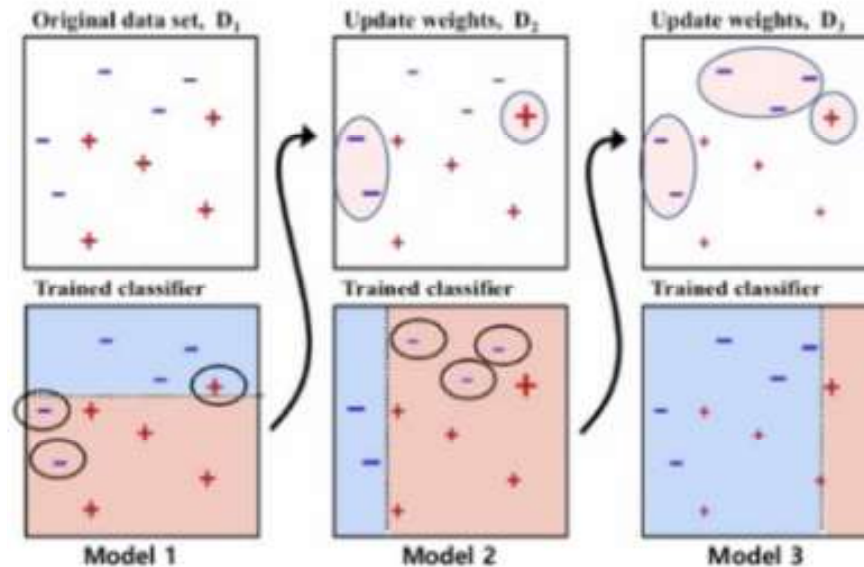
64	66	69	130	190	196	203	260
73	130	187	260	333	390	446	520
90	194	297	390	484	584	683	780
130	260	390	520	650	780	910	1040
162	326	491	650	808	976	1145	1300
203	390	577	780	983	1170	1357	1560
252	454	655	910	1166	1364	1561	1820
260	520	780	1040	1300	1560	1820	2080

Integral Image

Viola Jones

- AdaBoost

- Ensemble은 weak learner를 이용하여 strong learner로 결합할 수 있는 기법인데, Boosting은 Ensemble의 한 종류이다.
- Boosting은 앞 모델을 개선해나가면서 학습시키는 방법입니다.



Viola Jones

- Given example images $(x_1, y_1), \dots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \dots, T$:

1. Normalize the weights,

$$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}}$$

so that w_t is a probability distribution.

2. For each feature, j , train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t , $\epsilon_j = \sum_i w_i |h_j(x_i) - y_i|$.
3. Choose the classifier, h_t , with the lowest error ϵ_t .

4. Update the weights:

$$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.

- The final strong classifier is:

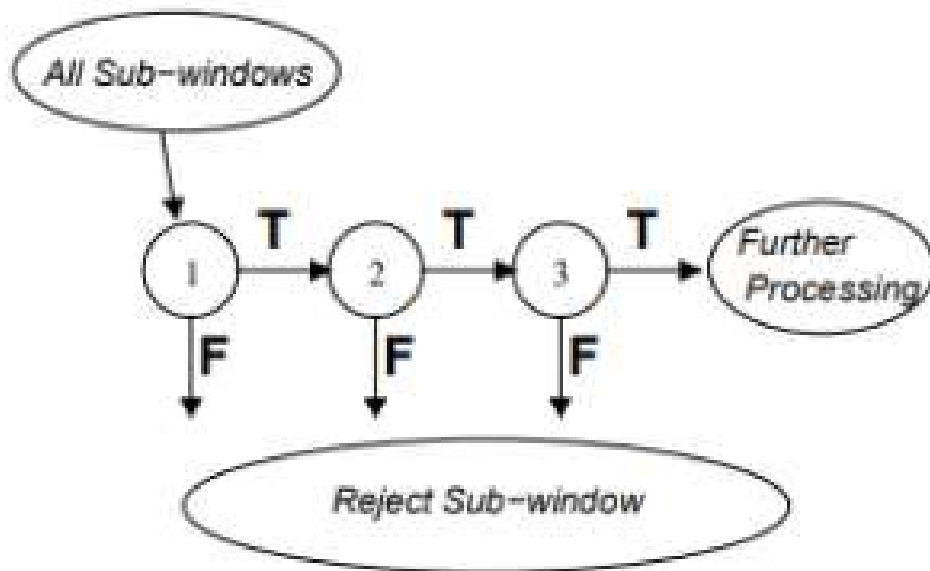
$$h(x) = \begin{cases} 1 & \sum_{t=1}^T \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

0 : 배경, 1 : 사람

Viola Jones

- Cascading Classifier



strong learner를 이용하여 물체를 검출할 때 weak learner를 많이 이용하는데 매번 사용한다면 real-time으로 출력 불가

따라서 Cascade 구조 제안

각 단계별로 물체로 분류되는 애들만 통과시키고 배경으로 분류되는 애들은 버림

$$h(x) = \begin{cases} 1 & \text{if } \sum_{t=1}^T \alpha_t h_t(x) \geq \tau \sum_{t=1}^T \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

임계값이 0.5보다 작은 값을 가져야만 높은 검출율을 얻을 수 있다.

Viola Jones

- Training

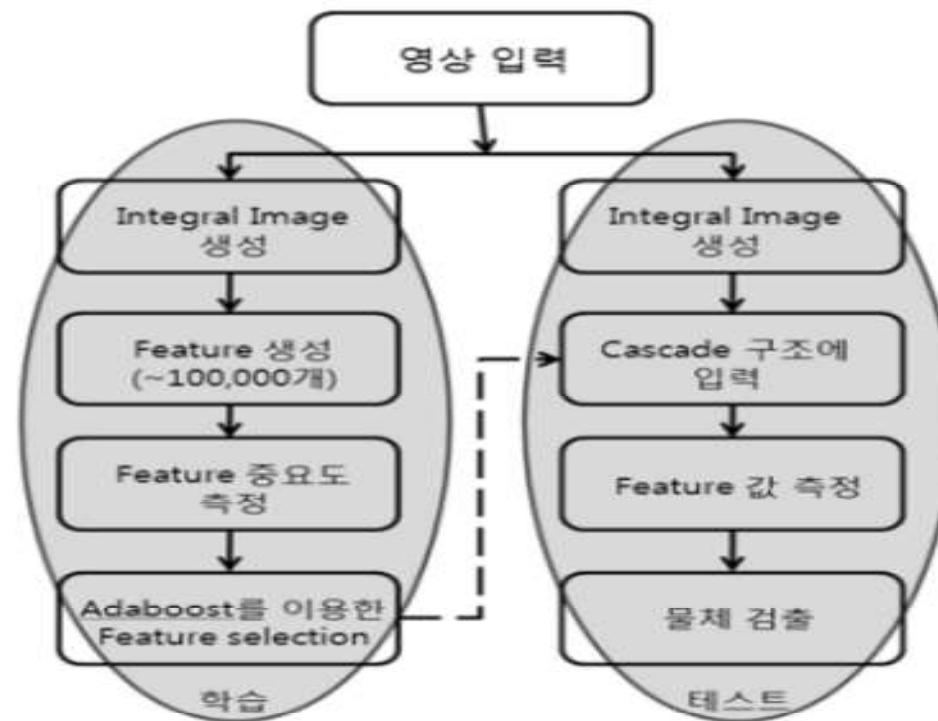


그림 5. Viola-Jones 알고리즘을 이용한 물체 검출 과정
Fig. 5. The process of object detection using Viola-Jones.

Viola Jones

- Results

- 38 layer의 Cascaded classifier는 얼굴 정면을 detect 하도록 train
- train data : face(4916개)x2,
non face(9544개)



Figure 5: Example of frontal upright face images used for training.

Viola Jones

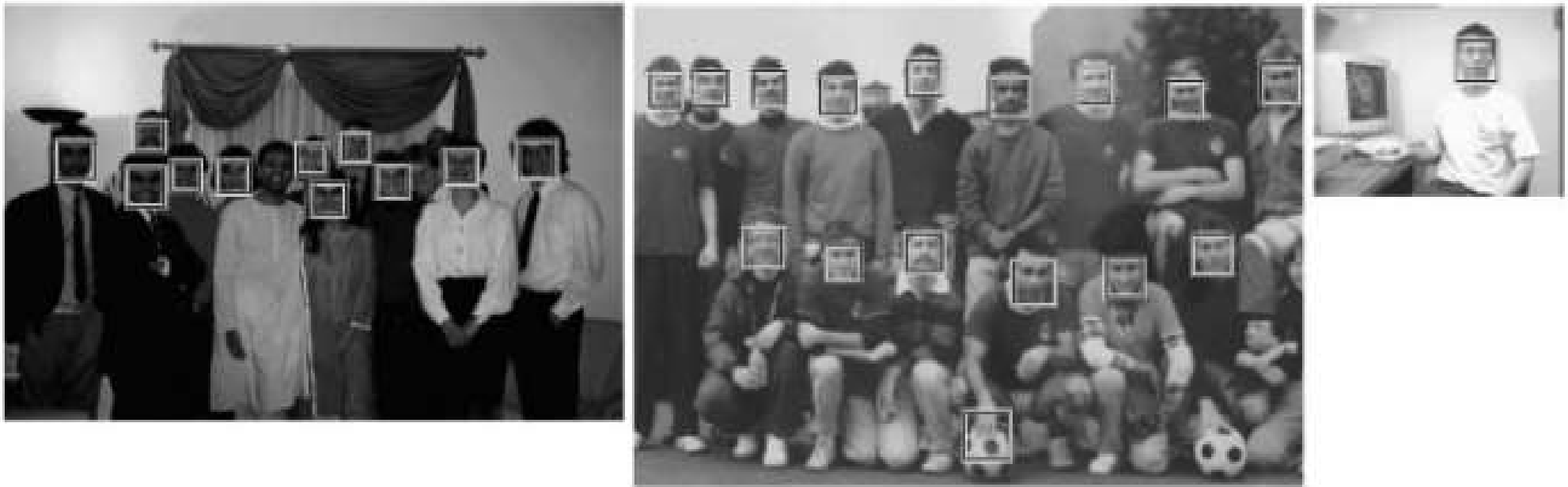


Figure 7: Output of our face detector on a number of test images from the MIT+CMU test set.

Viola Jones 실습 - OpenCV

```
# Import the library modules
import cv2
import sys

# Haar Feature XML 파일
cascPath = "haarcascade_frontalcatface.xml"
faceCascade = cv2.CascadeClassifier(cascPath)

# 컴퓨터와 연결된 0번 카메라 준비
video_capture = cv2.VideoCapture(0)

while True:
    # 프레임당 화면 캡처
    ret, frame = video_capture.read()

    # 컬러 GRAY로 변경
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # 파라미터 변경 가능
    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.05,
        minNeighbors=5,
        minSize=(100, 100),
        flags=cv2.CASCADE_SCALE_IMAGE
    )
```

scale Factor : 각 이미지 배율에서 이미지 크기를 줄임
minNeighbors : 사각형이 유지해야 하는 이웃 수??
minSize : 최소 객체 크기
flags : 이전 단계에 대해 동일한 의미를 가짐

Viola Jones 실습 - OpenCV

```
# 얼굴에 사각형 그려줄
for (x, y, w, h) in faces:
    cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

# 프레임 보기
cv2.imshow('Video', frame)

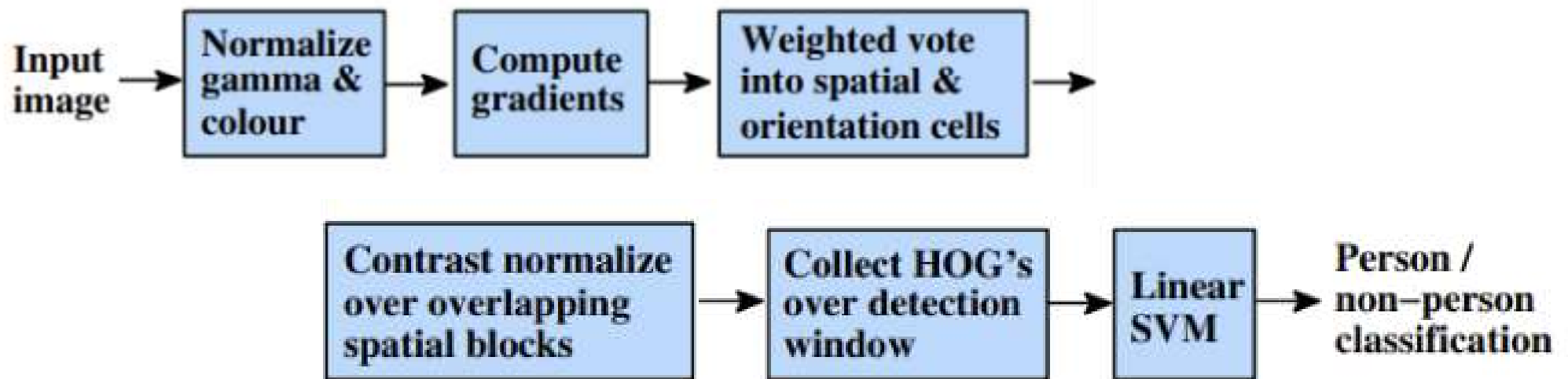
# 프로그램 종료하려면 Q 누르면 된다.

if cv2.waitKey(1) & 0xFF == ord('q'):
    break

# 프로그램 종료, 캡처 해제
video_capture.release()
cv2.destroyAllWindows()
```

HOG

- HOG(Histogram of Oriented Gradient)
 - 방향 그레디언트 히스토그램을 이용하여 Feature vector를 정의하고 SVM을 이용하여 위치를 Detection하는 알고리즘



HOG

- Abstract

- Linear SVM을 기반으로 보행자 Detection을 적용한 visual Object Recognition을 위한 Feature set에 대하여 작성
- descriptor를 기반으로 edge와 그레디언트를 계산한 후, descriptor가 보행자 Detection을 위한 Feature set에 잘 맞는 것을 실험적으로 보인다.
- 좋은 결과를 위해 중간 과정들이 필수적이라는 결과를 내는 연산의 영향을 연구한다.

HOG

- Datasets

1) well-established MIT pedestrian database
(train 509, test 200 images)(left-right reflection)

-> pose : front or back views

2) INRIA (data 1805 by 64 x 128)

-> pose : people standing



HOG

i) Normalized gamma & colour

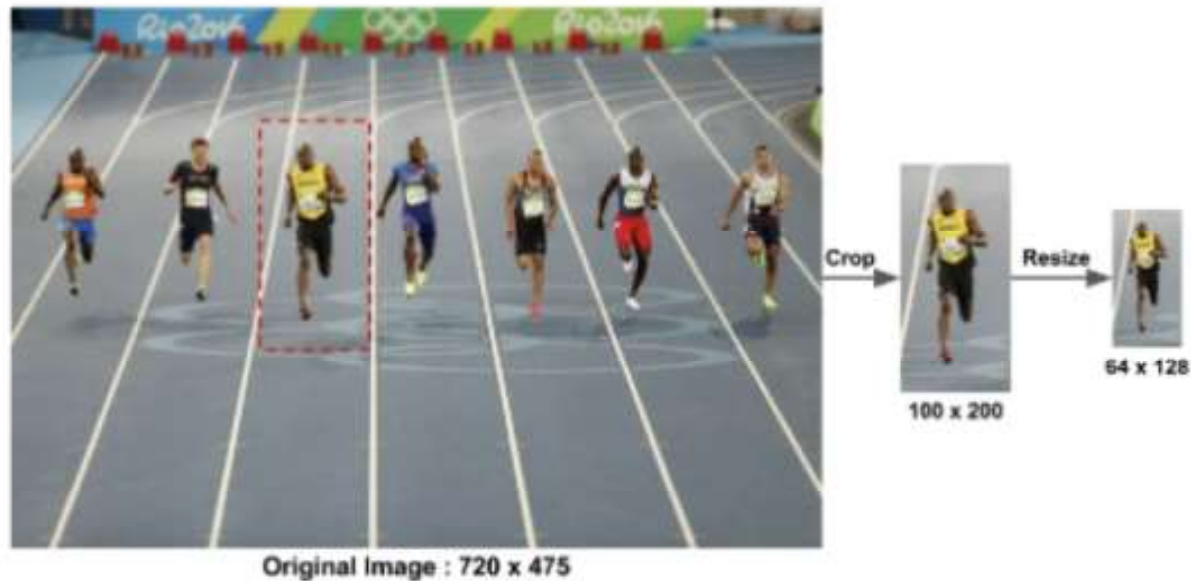
- gamma값 보정을 의미한다.
- 사진에서 gamma를 움직이는 것은 밝기 또는 휘도를 보정한다고 생각하면 된다.
- 하지만 gamma 보정은 효과가 미미하기 때문에 안해도 된다.

$$V_{\text{out}} = AV_{\text{in}}^{\gamma}$$

HOG

ii) Image Resize

- HOG descriptor는 64*128 Image에서 계산이 된다.
- Image의 size는 다를 수 있으나 비율을 1:2로 맞춰야 한다.



HOG

iii) Gradient Computation

- 수평 & 수직 변화량을 계산해야한다.
- 즉, x 방향으로 픽셀 값 차이와 y 방향으로의 픽셀값 차이를 구해야한다.
- HOG 논문에서는 실험 결과 $[-1,0,1]$ kernel에서 좋은 결과가 나왔다고 한다.

HOG

152	68	125
170	85	99
214	56	200

Gradient 계산



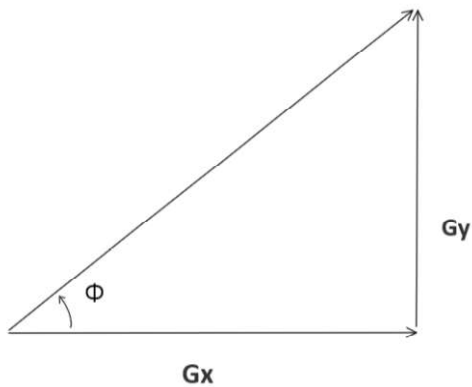
	0	
0		11
	-14	

HOG

iii) Gradient Computation

- 크기와 방향을 구해야 한다.

1) 크기



크기는 변화량이므로

$$\sqrt{(G_x)^2 + (G_y)^2}$$

아까 데이터에 대입하면
17.8이 나온다.
참고로 루트 없이 계산한
크기가 더 성능이 좋다고 한다.

2) 방향

방향은 역삼각함수인 arctan함수를 사용

$$\tan(\phi) = \frac{G_y}{G_x}$$

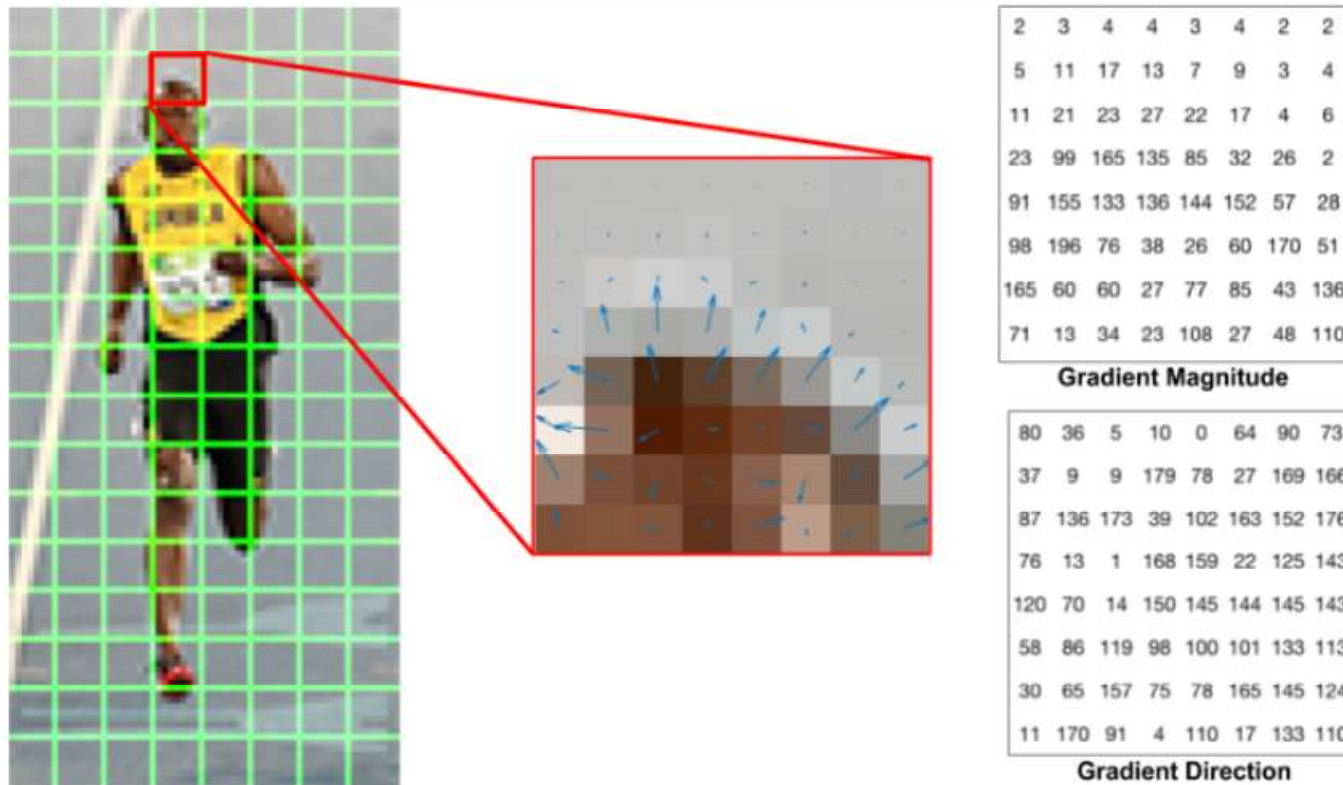


$$\phi = \arctan \frac{G_y}{G_x}$$

아까 데이터에 대입하면
-51.8이 나온다.

HOG

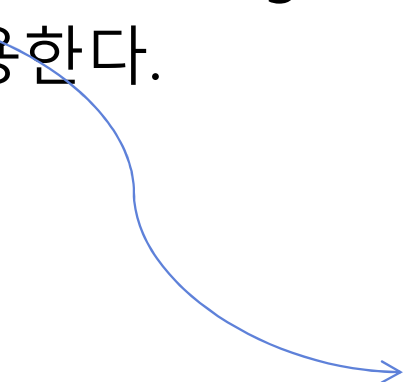
iv) Weighted vote into spatial & orientation cells



HOG

iv) Weighted vote into spatial & orientation cells

- 방향을 기준으로 구간(bin)을 나눠줄 것이다.
- bin이 9개일 때 성능이 많이 향상된다. 0~180도
- 위 단계에서 8x8 cell로 나누고 gradient histogram 계산을 한다.
- 이때 vote 특성을 이용한다.

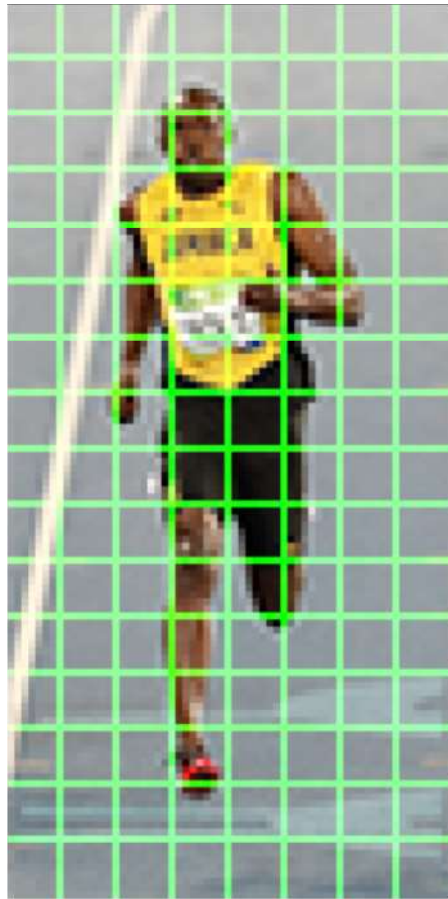


더 간단한 표현 가능,
노이즈에 덜 민감,
원하는 Feature를 찾기 충분한 size

HOG

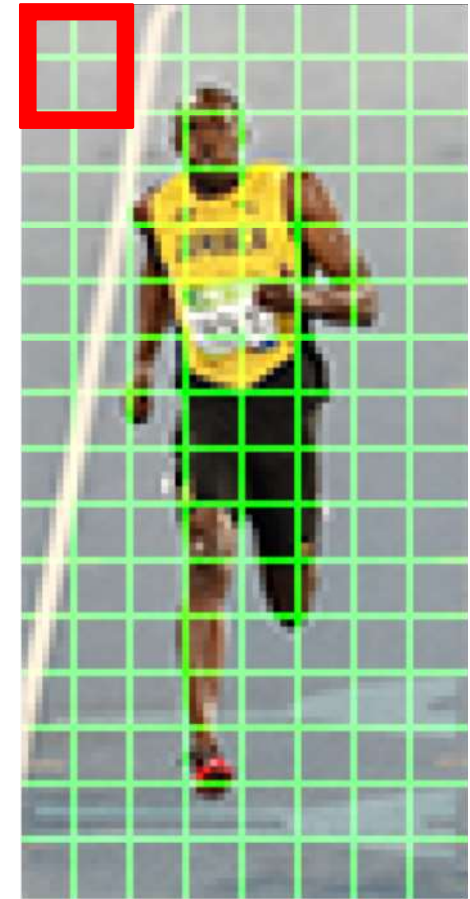
- Cell

8x8 로 나뉜
pixels

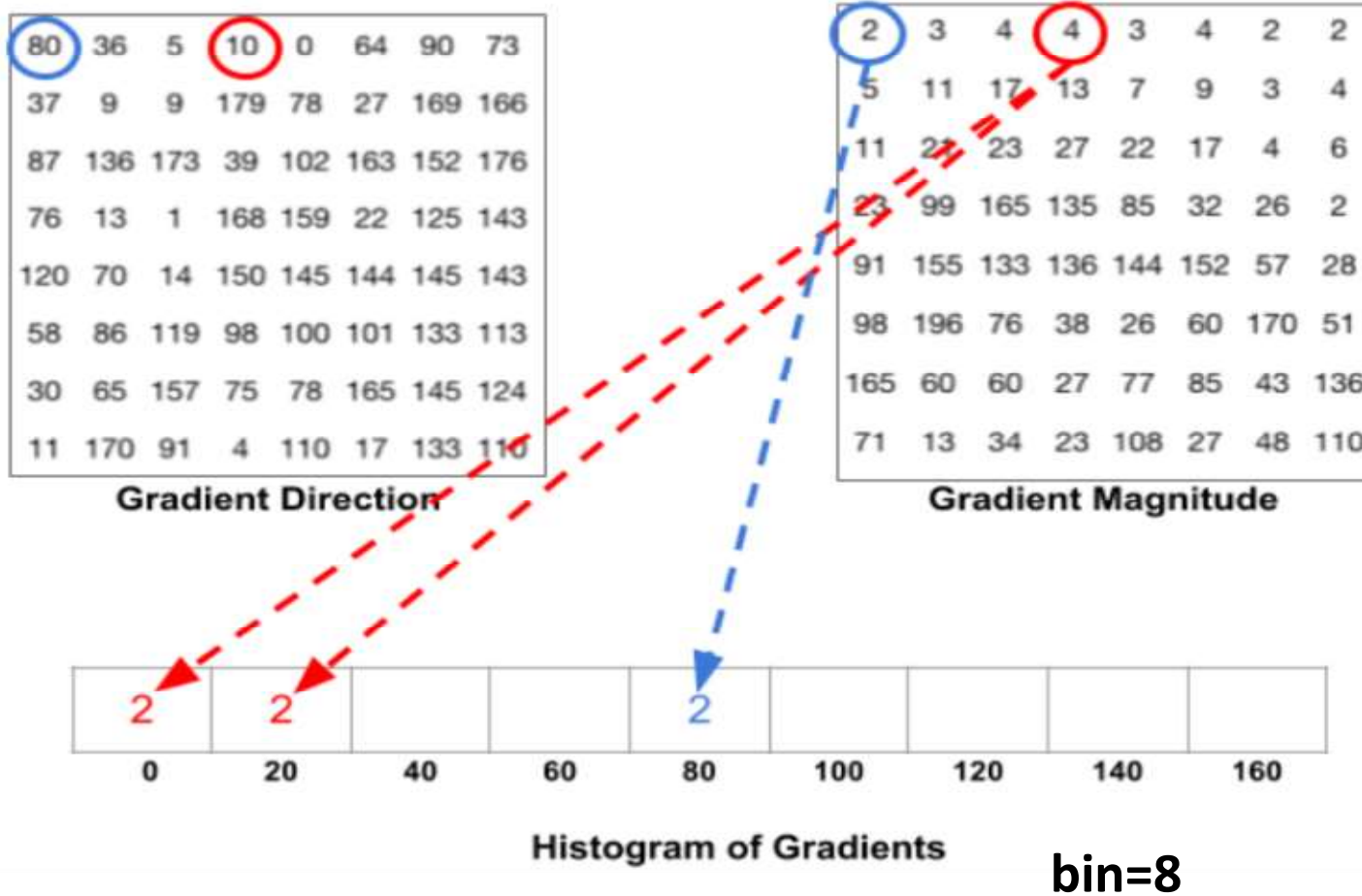


- Block

cell들의 집합



HOG



한 이미지에 대하여
크기와 방향을 구할 수 있다.

특정 픽셀이
방향 80 크기 : 2이면
bin 범위 중 80에 해당하는
계급 구간에 2를 채워넣음

방향 : 10 크기 4이면
10은 0과 20 사이이므로 각각
2를 채워넣음

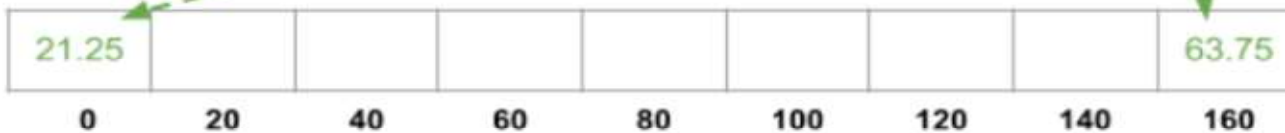
HOG

80	36	5	10	0	64	90	73
37	9	9	179	78	27	169	166
87	136	173	39	102	163	152	176
76	13	1	168	159	22	125	143
120	70	14	150	145	144	145	143
58	86	119	98	100	101	133	113
30	65	157	75	78	165	145	124
11	170	91	4	110	17	133	110

Gradient Direction

2	3	4	4	3	4	2	2
5	11	17	13	7	9	3	4
11	21	23	27	22	17	4	6
23	99	165	135	85	32	26	2
91	155	133	136	144	152	57	28
98	196	76	38	26	60	170	51
165	60	60	27	77	85	43	136
71	13	34	23	108	27	48	110

Gradient Magnitude

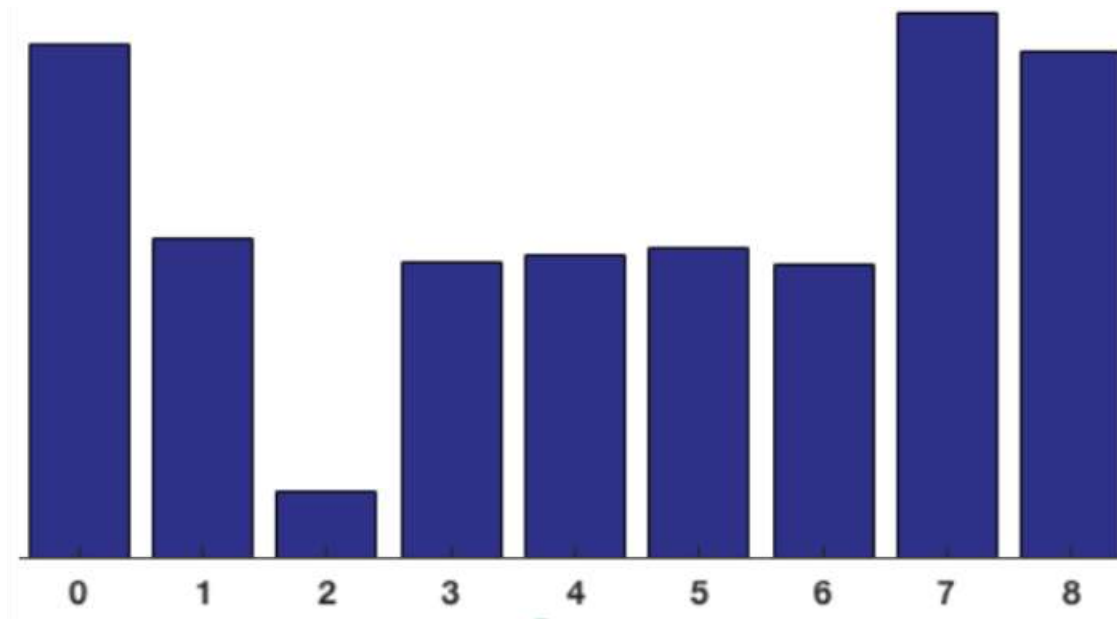


Histogram of Gradients

한 이미지에 대하여
크기와 방향을 구할 수 있다.

특정 픽셀이
방향 : 165 크기 : 85이면
160과 180 사이므로
가중치를 반영하여
다음과 같이
63.75과 21.25를 채워넣음.

HOG

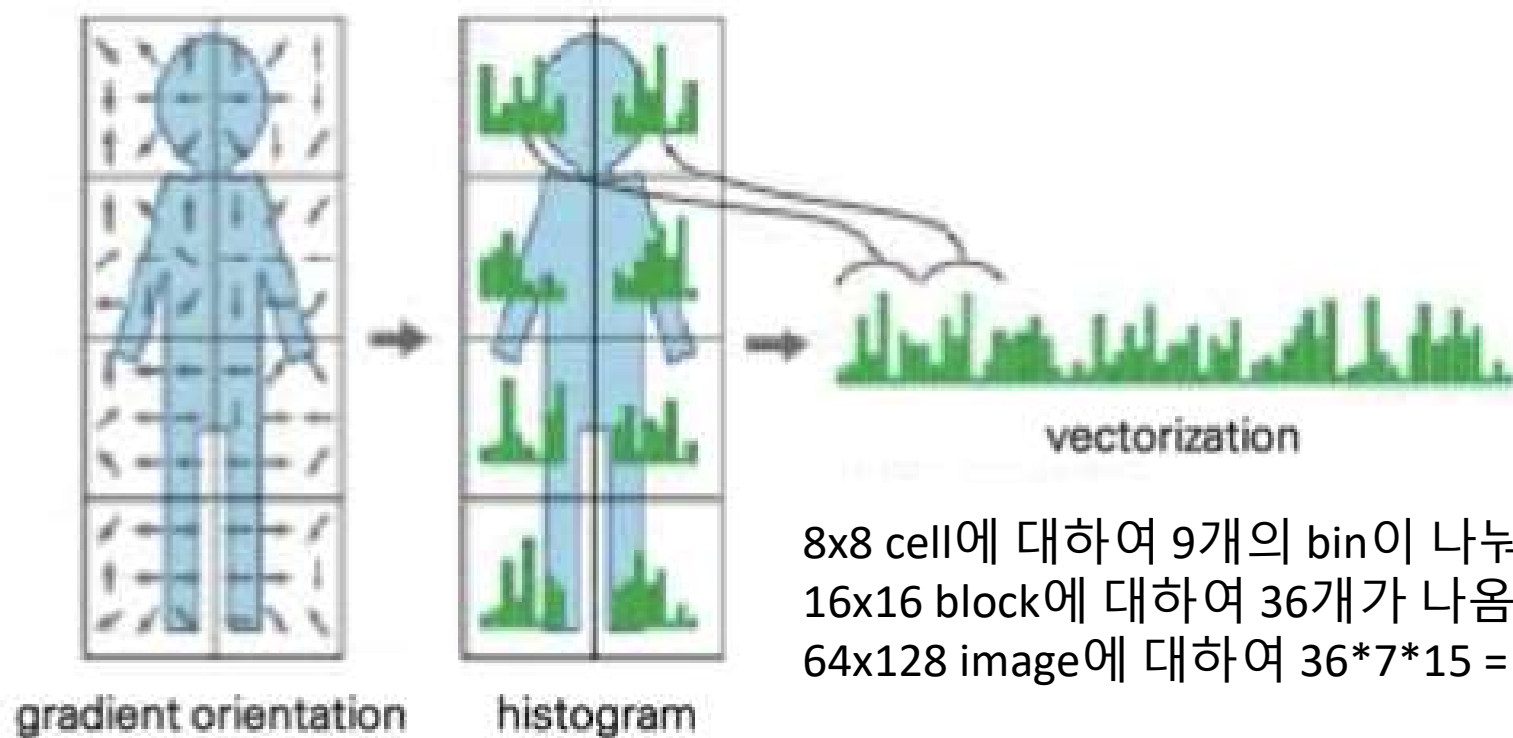


이를 Histogram화 시키면 다음과 같다.

0도, 180도 근처에서
많은 가중치를 갖는 것을 볼 수 있다.

이는 방향이 위 또는 아래를 많이
가리키고 있다는 뜻!

HOG



8x8 cell에 대하여 9개의 bin이 나뉘짐
16x16 block에 대하여 36개가 나옴
64x128 image에 대하여 $36 \times 7 \times 15 = 3780$ vector를 가짐

HOG

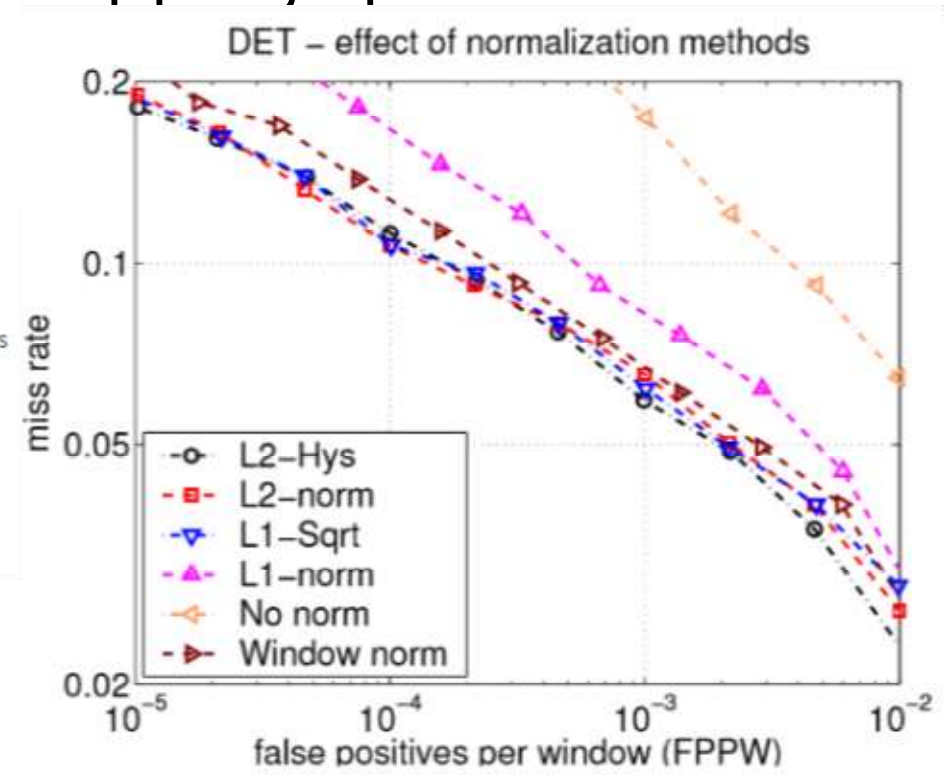
v) Contrast normalize over overlapping spatial blocks

$$\text{L2-norm: } f = \frac{v}{\sqrt{\|v\|_2^2 + e^2}}$$

L2-hys: L2-norm followed by clipping (limiting the maximum values of v to 0.2) and renormalizing, as in [5]

$$\text{L1-norm: } f = \frac{v}{(\|v\|_1 + e)}$$

$$\text{L1-sqrt: } f = \sqrt{\frac{v}{(\|v\|_1 + e)}}$$



HOG

vi) Classifier

- soft liner SVM, SVMlight로 훈련
- 가우시안 kernel SVM을 이용하면 런타임을 늘어나지만 성능이 3% 향상되었다.

Reference

- [1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, vol. 1. IEEE, 2001, pp. I-I. : <https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf>
- [2] Viola-Jones 얼굴 인식 알고리즘 : <https://ichi.pro/ko/viola-jones-eolgul-insig-algolijeum-68928056709058>
- [3] Viola Jones 알고리즘 및 Haar 캐스케이드 분류기 : <https://ichi.pro/ko/viola-jones-algolijeum-mich-haar-kaeseukeideu-bunlyugi-261941383264216>
- [4] OpenCV XML : <https://github.com/opencv/opencv>
- [5] 논문 : <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1467360>
- [6] 13.3 HOG 알고리즘과 보행자 검출 : <https://thebook.io/006939/ch13/03/>
- [7] HOG : <https://donghwa-kim.github.io/hog.html>
- [8] HOG : <https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=tommybee&logNo=221173056260>
- [9] <https://learnopencv.com/histogram-of-oriented-gradients/>
- [10] <https://studyingfox.tistory.com/7?category=849140>
- [11] <https://donghwa-kim.github.io/hog.html>
- [12] <https://eehoeskrap.tistory.com/98>