Computer Vision and Deep Learning

Ch 05

강명묵

Contents

발상

Correspond problem : 두 영상에서 feature point를 추출하고 매칭을 통해 해당하는 특징점 쌍을 찾는 것.

Local feature : 좁은 지역을 보고 특징점 여부 판정

Local feature criteria (1)

1. Repeatability

같은 물체가 서로 다른 두 영상에 나타났을 때, 첫 번째 영상에서 검출된 feature point가
두 번째 영상에서도 같은 위치에서 높은 확률로 검출되어야 한다.

2. Invariance

물체에 이동, 회전, 스케일 변환이 발생해도 feature descriptor의 값은 비슷해야 한다.

3. Discriminative power

물체의 다른 곳에서 추출된 특징과 두드러지게 달라야 한다.
- 그렇지 않으면, 물체의 다른 곳에서 추출된 특징과 매칭될 위험 있음.

발상

Local feature criteria (2)

4. Locality
작은 영역을 중심으로 특징 벡터를 추출해야 물체에 occlusion이 발생해도 매칭이 안정적으로 동작한다.

5. Appropriate Quantity
물체에 이동, 회전, 스케일 변환이 발생해도 feature descriptor의 값은 비슷해야 한다.

6. Computational Efficiency
실시간으로 처리해야 하는 경우 계산 시간에 중요한 응용이 필수적이다.

위의 6개 조건은 **특징점 검출, 기술자 추출**의 구현에 있어서 만족해야 한다.

Local feature criteria는 때로 **Trade-off** 관계에 있음.
반복성을 높이면 **실시간 처리**가 느려짐
분별력을 높이면, **지역성**이 낮아짐

이동과 회전 불변한 지역 특징

Moravec Algorithm

Moravec Algorithm

SSD(Sum Of Squared Difference)를 사용

$$S(v,u) = \sum_{y}\sum_{x}(f(y+v, x+u) - f(y,x))^2$$

original img(y,x)에서, 위의 식을 통해 **S-map(v,u)**을 추출

	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	c 0	0	0
2	0	0	0	1	0	0	0	0	0	0
3	0	0	0	1	1	0	0	0	0	0
4	0	0	0	b 1	1	1	0	0	0	0
5	0	0	0	1	1	1	1	0	0	0
6	0	0	0	1	1	1	1	a 1	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

Original Image

S-map의 픽셀 값에 따른 특징

v \ u	-1	0	1
-1	3	4	4
0	2	0	2
1	4	3	2

(a)

v \ u	-1	0	1
-1	3	1	6
0	3	0	4
1	3	0	3

(b)

v \ u	-1	0	1
-1	0	0	0
0	0	0	0
1	0	0	0

(c)

(a) : 코너

(b) : 경계

(c) : 주변에 화소의 변화 x

이동과 회전 불변한 지역 특징

Moravec Algorithm

특징 가능성 값 C

$$C = min(S(0,1), S(0,-1), S(1,0), S(-1,0))$$

- S-map을 이용해 지역 특징으로 좋은 정도를 측정
- C의 값이 클 수록 좋은 점수(모든 방향으로 높은 값을 가지기 때문에)

Moravec Algorithm의 한계점

- 한 화소에 대해 네 방향만 보고 특징점의 여부를 계산(3x3 마스크 쓰기 때문)
- 잡음에 대한 대처 방안 없음

이동과 회전 불변한 지역 특징

Harris feature point

Harris feature point

잡음에 대처하기 위해 **가우시안**을 추가

$$S(v,u) = \sum_y \sum_x G(y,x)(f(y+v,x+u) - f(y,x))^2 \qquad (1.1)$$

Taylor Expension

$$f(y+v,x+u) \approx f(y,x) + vd_y(y,x) + ud_x(y,x) \qquad (1.2)$$

(1.2)를 (1.1)에 대입

$$\boxed{S(v,u) \approx \sum_y \sum_x G(y,x)(vd_y(y,x) + ud_x(y,x))^2}$$

$$S(v,u) \approx \sum_y \sum_x G(y,x)(vd_y(y,x) + ud_x(y,x))^2$$

$$S(v,u) \approx \sum_y \sum_x G(y,x)(v^2d_y^2 + 2vud_yd_x + u^2d_x^2)$$

$$\begin{aligned} S(v,u) &\approx \sum_y \sum_x G(y,x)(v\ \ u)\begin{bmatrix} d_y^2 & d_yd_x \\ d_yd_x & d_x^2 \end{bmatrix}\begin{bmatrix} v \\ u \end{bmatrix} \\ &= (v\ \ u)\sum_y \sum_x G(y,x)\begin{bmatrix} d_y^2 & d_yd_x \\ d_yd_x & d_x^2 \end{bmatrix}\begin{bmatrix} v \\ u \end{bmatrix} \\ &= (v\ \ u)\begin{bmatrix} \sum_y\sum_x G(y,x)d_y^2 & \sum_y\sum_x G(y,x)d_yd_x \\ \sum_y\sum_x G(y,x)d_yd_x & \sum_y\sum_x G(y,x)d_x^2 \end{bmatrix}\begin{bmatrix} v \\ u \end{bmatrix} \end{aligned}$$

$$S(v,u) = (v\ \ u)\begin{bmatrix} G * d_y^2 & G * d_yd_x \\ G * d_yd_x & G * d_x^2 \end{bmatrix}\begin{bmatrix} v \\ u \end{bmatrix} = uAu^T \qquad (1.3)$$

Second moment matrix

$$A = \begin{bmatrix} G * d_y^2 & G * d_yd_x \\ G * d_yd_x & G * d_x^2 \end{bmatrix}$$

- S-map을 구할 때 정수 뿐 아니라 **실수**도 가능
- A만 분석하면 **지역 특징 여부** 판단 가능
- A의 **고윳값**을 통해 지역 특징으로 좋은 정도 측정 가능

이동과 회전 불변한 지역 특징

Harris feature point

특징 가능성 값 C

$$C = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \qquad (1.5)$$

- **K는 0.04**로 설정하는 것이 적절

$$Let \quad A = \begin{bmatrix} p & r \\ r & q \end{bmatrix} (8)$$

$$\lambda_1 + \lambda_2 = p + q, \quad \lambda_1 \lambda_2 = pq - r^2$$

$$C = (pq - r^2) - k(p+q)^2 \qquad (1.6)$$

고윳값의 크기에 따른 특징

1. λ가 모두 0 :
특징으로서 가치 x

2. λ 가 하나만 큼 :
한 방향으로만 변화 있음

3. λ가 모두 큼 :
지역 특징으로써 매우 적합

(1.5)의 식은 λ값을 구해야 하는 계산이 필요하기 때문에,
계산 식을 줄이기 위해서 (1.6)의 식을 이용하여 코드 작성

$$Let\ A = \begin{bmatrix} 0.52 & -0.2 \\ -0.2 & 0.53 \end{bmatrix}$$

$$\rightarrow \lambda I - A = \begin{bmatrix} \lambda - 0.52 & 0.2 \\ 0.2 & \lambda - 0.53 \end{bmatrix}$$

$$\det(\lambda I - A) = \begin{vmatrix} \lambda - 0.52 & 0.2 \\ 0.2 & \lambda - 0.53 \end{vmatrix} = 0$$

$$(\lambda - 0.52)(\lambda - 0.53) - 0.04 = 0$$

$$\lambda^2 - 1.05\lambda + 0.2356 = 0$$

$$\lambda_1 = 0.725, \quad \lambda_2 = 0.325$$

$$C = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2 \leftarrow (k = 0.04)$$

$$= 0.1915$$

이동과 회전 불변한 지역 특징

Harris feature point

Localization

•특징 가능성 맵 C에서 특징일 가능성이 가장 높은 곳을 선택해야 함
•extreme point를 찾기 위해 **non maximum suppression** 사용
- 주변 8개의 이웃보다 큰 값을 갖는 화소를 **feature point**로 선택

이동과 회전 불변한 지역 특징

Harris feature point

Harris feature point detection Algorithm
식 두 개를 활용하여 코드 구현

$$A = \begin{bmatrix} G * d_y^2 & G * d_y d_x \\ G * d_y d_x & G * d_x^2 \end{bmatrix}$$

$$Let \quad A = \begin{bmatrix} p & r \\ r & q \end{bmatrix}$$

$$C = (pq - r^2) - k(p+q)^2$$

Algorithm

1. 특징 가능성 맵 찾기
   - 10x10 matrix 생성
   - ux, uy vector 생성 [-1,0,1], [-1,0,1].T
   - 3x3(σ = 1) Gaussian 생성 (g)
   - img와 uy 컨볼루션 (dy)
   - img와 uv 컨볼루션 (dx)
   - dyy(dy*dy)와 g 컨볼루션 (gdyy)
   - C = (gdyy*gdxx - gdyx*gdyx) - 0.04*(gdyy+gdxx)*(gdyy+gdxx)

2. 비최대 억제를 사용하여 특징점 찾기
   - 특징점은 9로 표시

parameter

- σ(gaussian) = 1

이동과 회전 불변한 지역 특징

Harris feature point

Harris feature point detection Algorithm 1
특징 가능성 맵 C 찾기

```
//step 1 : 10x10 img 생성
Mat img = (Mat_<double>(10, 10) <<
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
    0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
    0, 0, 0, 1, 1, 1, 1, 0, 0, 0,
    0, 0, 0, 1, 1, 1, 1, 1, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0);

//step2
Mat ux = (Mat_<double>(1, 3) << -1, 0, 1);
Mat uy = ux.t();
Mat dy = conv_img(img, uy);
Mat dx = conv_img(img, ux);
Mat dyy = dy.mul(dy);
Mat dyx = dy.mul(dx);
Mat dxx = dx.mul(dx);

//step3 Gaussian filtering
Mat g = Gaussian_Kernel(3, 1);
Mat gdyy = conv_img(dyy, g);
Mat gdxx = conv_img(dxx, g);
Mat gdyx = conv_img(dyx, g);
print(g);

//step4 calculate C
Mat C = (gdyy.mul(gdxx) - gdyx.mul(gdyx)) - 0.04 * ((gdxx + gdyy)).mul((gdxx + gdyy));
```

1. 특징 가능성 맵 찾기
   - 10x10 matrix 생성
   - ux, uy vector 생성 [-1,0,1], [-1,0,1].T
   - 3x3(σ = 1) Gaussian 생성 (g)
   - img와 uy 컨볼루션 (dy)
   - img와 uv 컨볼루션 (dx)
   - dyy(dy*dy)와 g 컨볼루션 (gdyy)
   - C = (gdyy*gdxx - gdyx*gdyx) - 0.04*(gdyy+gdxx)*(gdyy+gdxx)

이동과 회전 불변한 지역 특징

Harris feature point

Harris feature point detection Algorithm 2

non maximum suppression을 사용하여 feature point 찾기

```
Mat non_maximum_suppression(Mat& img_origin, Mat& img_C) {
    Mat img_non_max = img_origin.clone();
    for (int y = 0; y < img_origin.rows; y++) {
        for (int x = 0; x < img_origin.cols; x++) {
            int count = 0;
            if (img_C.at<double>(y, x) > 0.1) {
                for(int m = -1; m < 2; m++){
                    for (int n = -1; n < 2; n++) {
                        if (img_C.at<double>(y, x) > img_C.at<double>(y + m, x + n)) count++;
                    }
                }
            }
            if (count == 8) img_non_max.at<double>(y, x) = 9;
        }
    }
    return img_non_max;
}
```

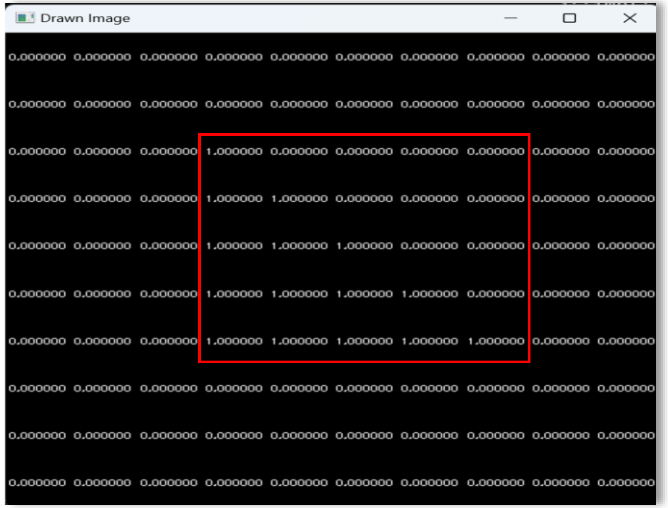
2. 비최대 억제를 사용하여 특징점 찾기
   - 특징점은 9로 표시

이동과 회전 불변한 지역 특징

Harris feature point

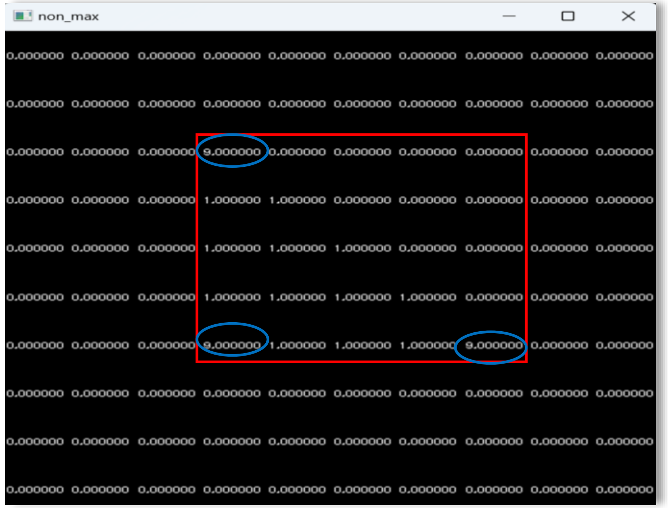
Conclusion

Origin image



특징 가능성 맵 C

non max suppression



결과값을 보면 세 특징점이 모두 모퉁이(corner)에 존재하지만, 실제 영상에 적용하면 **모퉁이(corner) 뿐만 아니라 블롭(blob)**도 많이 검출한다.
따라서 이렇게 검출된 값은 **feature point 또는 interest point** 라고 한다.

스케일 불변한 지역 특징

Scale space feature detection

Scale space feature detection Algorithm

1. 입력 영상 f로부터 다중 스케일 영상 $\tilde{f}$ 구성
2. $\tilde{f}$에 적절한 미분 연산을 적용하여 다중 스케일 미분 연산 $\tilde{f}'$를 구한다.
3. $\tilde{f}'$에서 극점을 찾아 특징점으로 취한다.

Input : grayscale image f
Output : scale-invariant feature points

Algorithm_1

1. Gaussian Smoothing
거리가 멀어지면 물체가 **흐려지는 것**을 묘사
표준편차 σ를 점점 키움

2. Pyramid
거리가 멀어지면 물체의 **크기가 작아지는 것**을 묘사
크기를 점점 줄임

Algorithm_2

스케일 공간의 미분
Laplacian을 주로 사용

$$\nabla^2 f = d_{yy} + d_{xx} \qquad (1.7)$$

$$\nabla^2_{normal} f = \sigma^2 |d_{yy} + d_{xx}| \qquad (1.8)$$

(1.7) Laplacian

(1.8) Normalized Laplacian

Algorithm_3

극점 검출
3차원에서의 **non maximum suppression** 사용

SIFT

Lowe 2004

1. Scale-space extrema detection
스케일 및 방향에 불변한 후보 지점을 식별하기 위해 Difference-of-Gaussian을 사용(Scale-invariant 달성)

2. Keypoint localization
위에서 구한 후보 지점에서 **세부 모델**을 적합하여 위치와 스케일 결정

3. Orientation assignment
각 키포인트 위치에 하나 이상의 방향을 local image gradient 방향을 기반으로 할당

4. Keypoint descriptor
형태 왜곡 및 조명 변화에 대한 상당한 수준의 지역적인 변형 허용하게끔 함.

SIFT

1. Scale-space extrema detection

1.1 Detection of scale-space extrema

Keypoint detection의 첫 번째 과정은 **locations**와 **scales**를 찾는 것이다.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

input image : I(x, y)

a variable-scale Gaussian: G(x, y, σ)

the scale space of an image : L(x, y, σ)

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned}$$

the difference-of-Gaussian function : D(x, y, σ)

constant multiplicative factor : k

(2.2)의 열 확산 방정식 원리와 (2.1)의 식을 이용하여 우측 상단 식 유도

$$\frac{\partial G}{\partial \sigma} = \sigma \nabla^2 G \qquad (2.1)$$

열 확산 방정식 $\frac{\partial T}{\partial t} = D\nabla^2 T \qquad (2.2)$

$$\sigma \nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y, \sigma)}{k\sigma - \sigma}$$

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k - 1)\sigma^2 \nabla^2 G$$

SIFT

1. Scale-space extrema detection

1.1 Detection of scale-space extrema

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k-1)\sigma^2\nabla^2 G$$

Difference-of-Gaussian

$$(k-1)\sigma^2\nabla^2 G$$

Scale-normalized Laplacian of Gaussian

$$\sigma^2\nabla^2 G$$

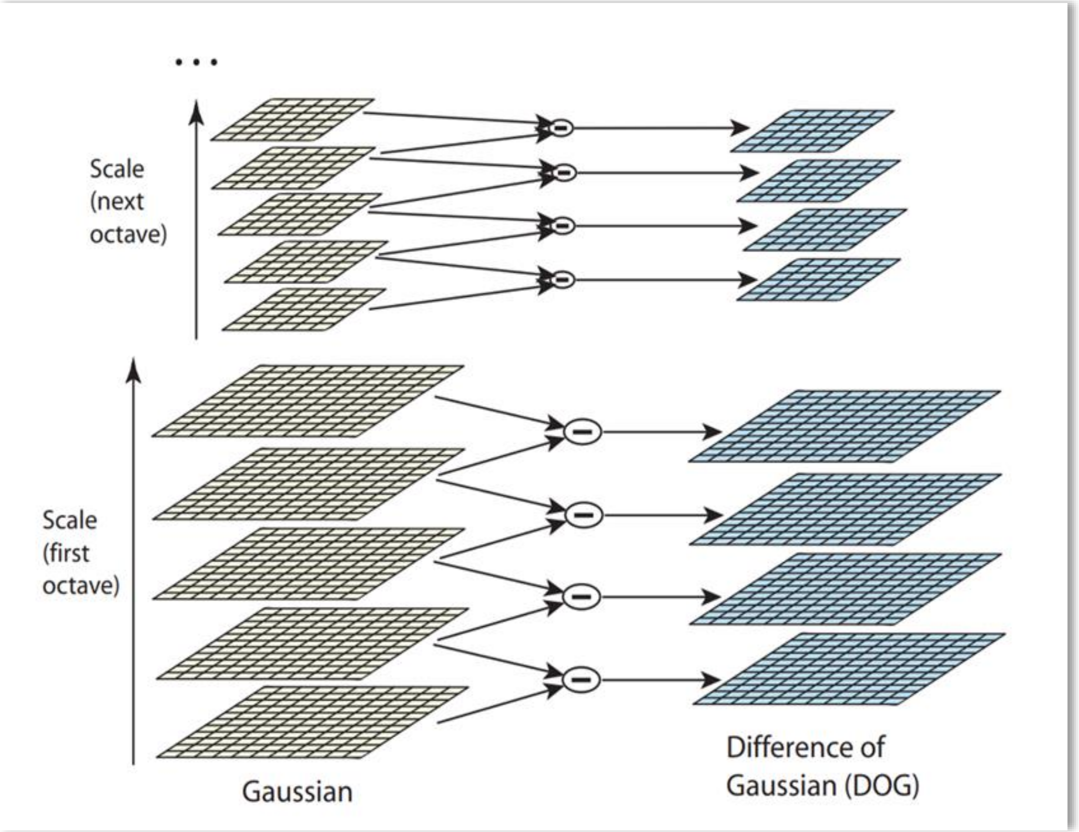
계산이 복잡한 **LoG를 DoG로 대체** 가능하다
- 즉, 단순한 **이미지 빼기 연산**으로 계산 가능
- (k-1)은 extrema loaction에 영향 x

SIFT

1. Scale-space extrema detection

1.1 Detection of scale-space extrema

다중 스케일 영상 구성



$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k-1)\sigma^2\nabla^2 G$$

$$k = 2^{\frac{1}{s}}$$

원본 이미지의 크기를 두 배로 확장하여 시작
각 octave당 s + 3개의 image를 생성

octave의 개수 : 3개
Sigma의 값 : 1.6
Image의 개수 : 6개

SIFT

1. Scale-space extrema detection

1.2 Local extrema detection

D(x,y,σ)의 extrema 찾기
maxima와 minima를 이웃의 26개의 image와 비교 하여 찾는다

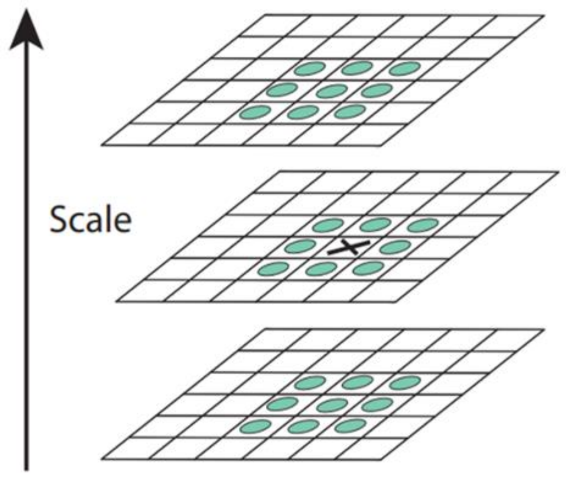


Figure 2: Maxima and minima of the difference-of-Gaussian images are detected by comparing a pixel (marked with X) to its 26 neighbors in 3x3 regions at the current and adjacent scales (marked with circles).

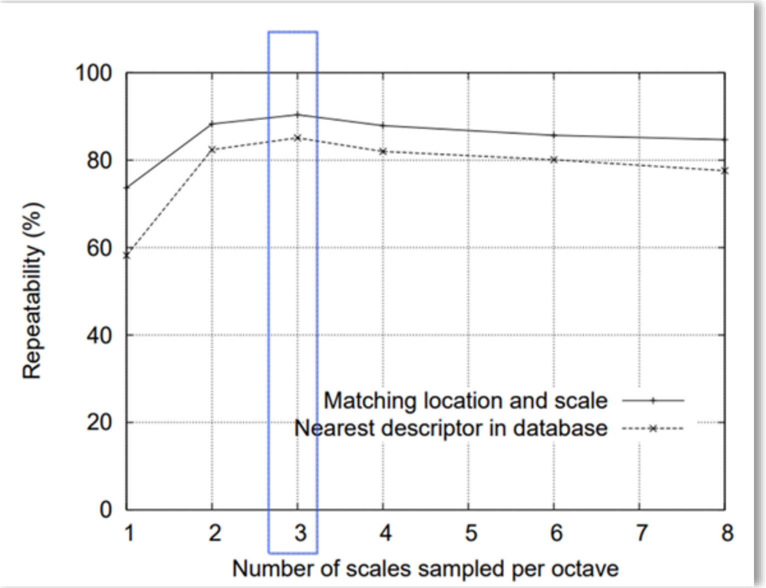
SIFT

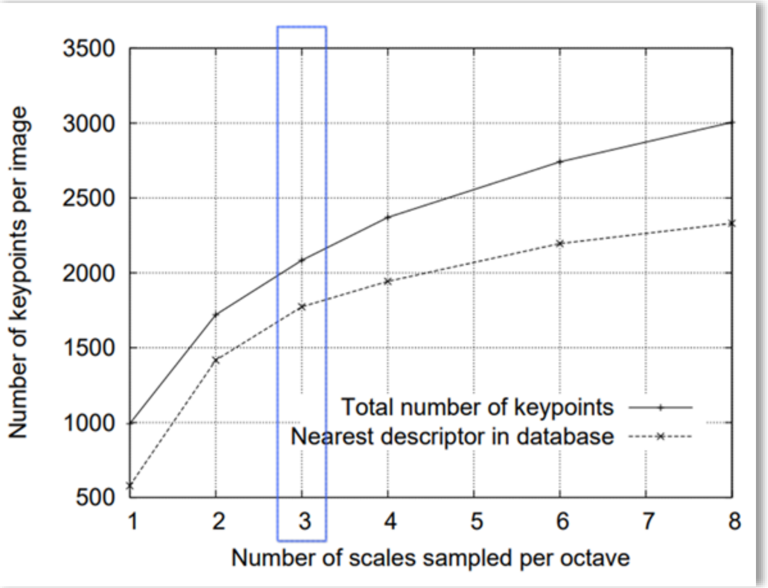
1. Scale-space extrema detection

1.3 Frequency of sampling in scale

샘플링 빈도 찾기(효율성과 안정성은 trade-off 관계)

- 샘플링을 많이 하면 안정성이 올라가나 효율성이 떨어짐
- 샘플링을 적게 하면 효율성이 올라가나 안정성이 떨어짐





octave가 3일때 재현율이 가장 높다.

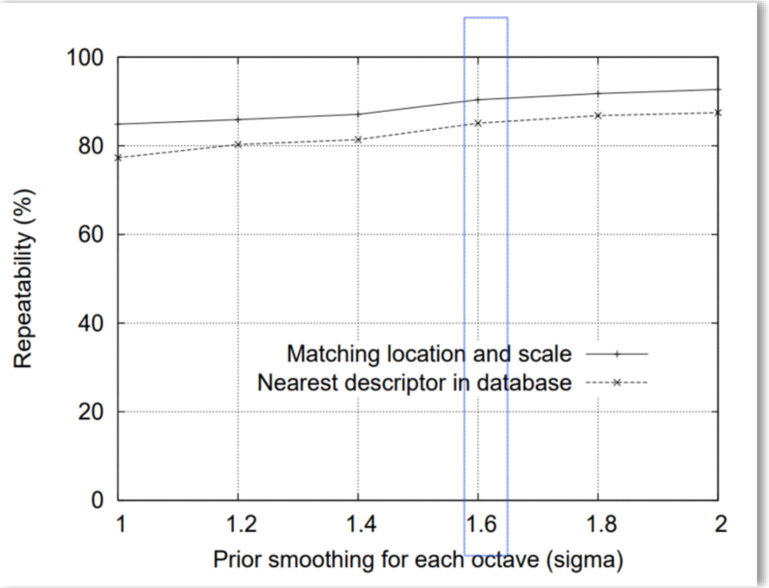
SIFT

1. Scale-space extrema detection

1.3 Frequency of sampling in the spatial domain

Sampling frequency과 rate of detection은 trade-off 관계

- σ값을 높이면 **repeatability** 올라가지만, cost도 같이 상승하여 **rate of detection**이 안 좋아짐.



sigma가 1.6일 때가 적절하다.

원본 이미지 σ는 0.5를 가지고 있다고 가정한다.

입력 이미지의 크기를 **2배**로 확장하면 **안정적인 키포인트 수**가 **거의 4배** 증가.

- σ = 1이 되어 추가적인 평활화는 필요 x
- 더 큰 확장계수는 유의미한 추가 향상 x

SIFT

1. Scale-space extrema detection

$\sigma_1 = 1.6$ $\quad \sigma_2 = 2.0159$ $\quad \sigma_3 = 2.5398$ $\quad \sigma_4 = 3.2000$ $\quad \sigma_5 = 4.0317$ $\quad \sigma_6 = 5.0797$

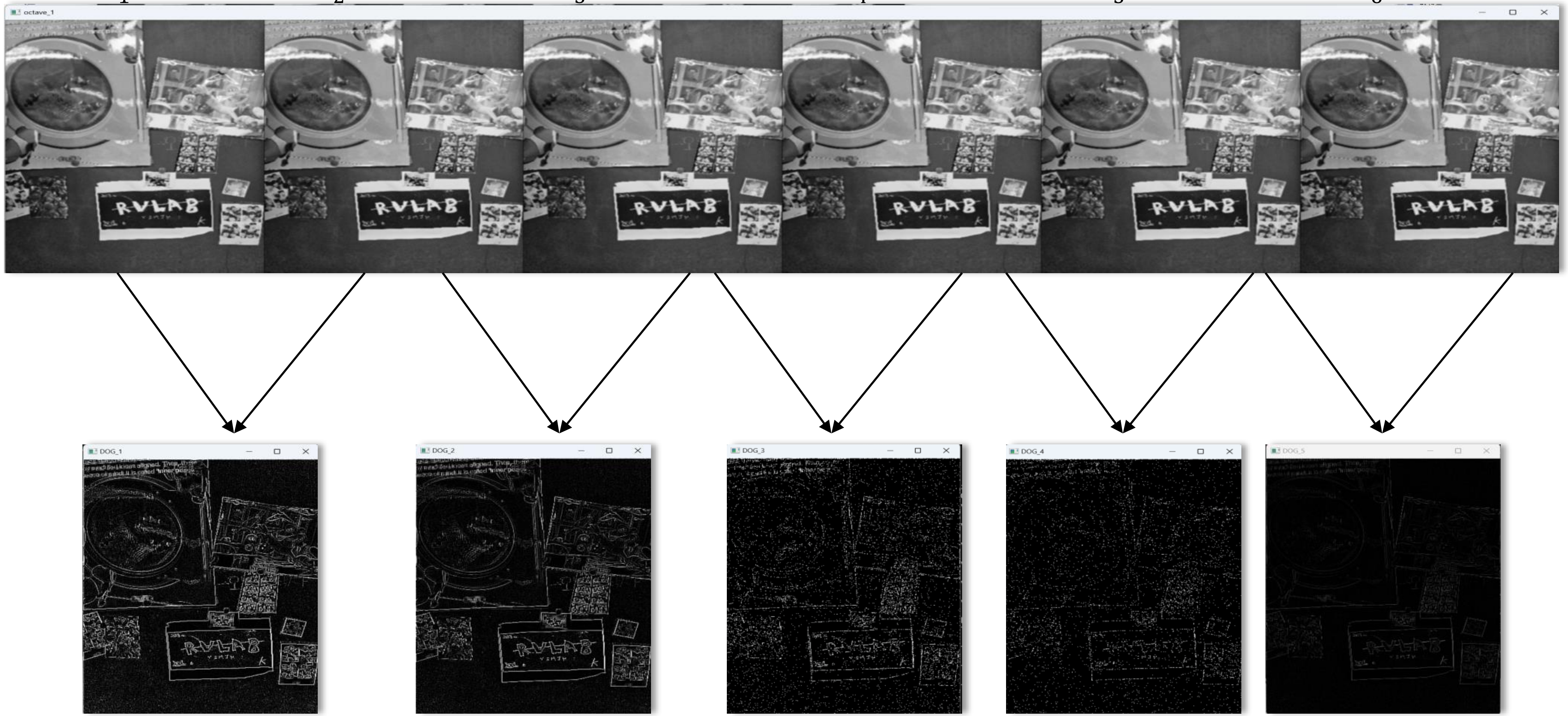
$$\sigma_{i+1} = 2^{\frac{1}{s}}\sigma_i$$

$s = 3$

SIFT

1. Scale-space extrema detection

$\sigma_1 = 1.6$ $\sigma_2 = 2.0159$ $\sigma_3 = 2.5398$ $\sigma_4 = 3.2000$ $\sigma_5 = 4.0317$ $\sigma_6 = 5.0797$



2. Keypoint localization

2.1 Accurate Keypoint localization

앞에서 찾은 키포인트 후보에 대해 **위치, 스케일 및 주요 곡률 비율**에 대한 **상세한 적합(fit)을 수행**
- **low contrast 제거**

Scale-space 공간 함수 **D(x,y,σ)**의 **Taylor expansion** 사용

$$D(x) = D + \frac{\partial D^T}{\partial x} + \frac{1}{2} x^T \frac{\partial^2 D}{\partial x^2} x \,, \quad x = (x, y, \sigma)^2$$

$$location\ of\ the\ extremum\ \hat{x} = -\frac{\partial^2 D^{-1}}{\partial x^2} \frac{\partial D}{\partial x}$$

$$D(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \hat{x} \qquad (2.3)$$

$$Discard\ if\ |D(\hat{x})| \leq 0.03 \qquad (2.4)$$
$$img\ pixel\ values\ in\ the\ range [0, 1]$$

The function value at the extremum (2.3)은 **low contrast에서의 불안정한 극점 제거**에 용이
- **Low contrast** : 특정 지역이나 물체가 주변과 비교했을 때 그 차이가 작아서 시각적인 구별 x

SIFT

2. Keypoint localization

2.2 Eliminating edge responses

stability를 위해서 **principal curvature를 통해 나머지 노이즈도 제거**

Hessian matrix H

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

H의 **고유값**들은 **D**의 **주된 곡률(principal curvature)과 비례**
Let $\alpha > \beta$

SIFT

2. Keypoint localization

$$Tr(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta$$

$$Det(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta$$

$$Det(\mathbf{H}) > 0 \qquad (2.5)$$

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r}$$

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} < \frac{(r+1)^2}{r} \qquad (2.6)$$

여기서

- 모든 고윳값이 **양수**인 경우 :
  - 위로 볼록 (함수는 극솟값을 가짐)
  - α >> β → Edge
- 모든 고윳값이 **음수**인 경우 :
  - 아래로 볼록 (함수는 극댓값을 가짐)
  - α << β → Edge
- 위 두 경우에 대해 α 와 β가 비슷하면 corner
- 고윳값의 부호가 다를 경우 :
  - 극값이 아니라는 것을 의미

즉, 아래 두 가지 조건을 시행한다.

1. Det(H)<0 인 경우 제거
   - 주된 곡률이 서로 다른 부호를 가지므로 해당 포인트는 극값이 아니라고 판단
2. r = 10의 값(논문에서 사용)을 사용하여 주된 곡률의 비율이 10보다 큰 키 포인트를 제거
   - r의 값이 클 수록 주된 곡률 간의 크기 차이가 크다는 것을 나타내고, 이는 안정성이나 신뢰성이 떨어질 가능성이 높다.

SIFT

2. Keypoint localization

(a)

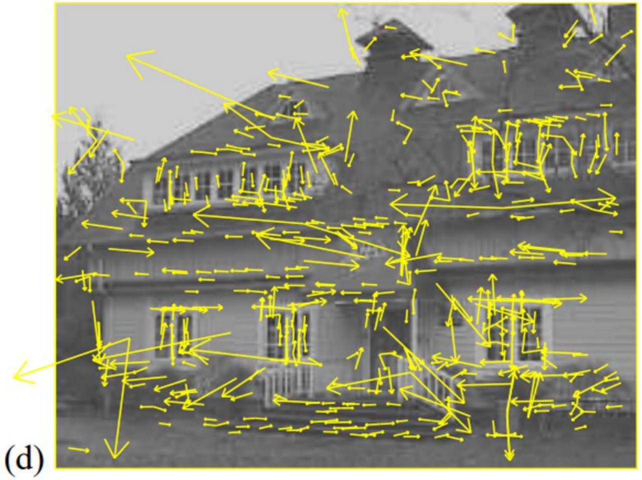
(a) The 233x189 pixel original image.

(b)

(b) The initial 832 keypoints **locations at maxima and minima of the difference-of-Gaussian function**. Keypoints are displayed as vectors indicating scale, orientation, and location.

(c)

(c) After applying a **threshold on minimum contrast**, 729 keypoints remain. - (2.4) 적용



(d)

The final 536 keypoints that remain following **an additional threshold on ratio of principal curvatures** - (2.5) (2.6) 적용

(a) Original image
(b) **DoG**를 사용한 후에 **extrema** 찾은 것
(c) **(2.4)** 적용
(d) **(2.5), (2.6)** 적용

$$Discard\ if\ |D(\hat{x})| \leq 0.03 \qquad (2.4)$$
$$img\ pixel\ values\ in\ the\ range[0,1]$$

$$Det(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta$$
$$Det(\mathbf{H}) > 0 \qquad (2.5)$$

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} < \frac{(r+1)^2}{r} \qquad (2.6)$$

r = 10

SIFT

2. Keypoint localization

SIFT

3. Orientation assignment

3. Orientation assignment

각 키포인트에 대해 **로컬 이미지 속성을 기반으로 일관된 방향** 할당함
-> **키포인트 기술자**는 이 방향을 기준으로 **상대적으로 표현**되어 **이미지 회전에 대한 불변성** 달성 가능

Algorithm

1. 각각의 **img sample L(x,y)**에 대해 **gradient magnitude m(x,y)와 orientation θ(x,y)** 계산

$$m(x,y) = \sqrt{((L(x+1,y) - L(x-1,y))^2 + ((L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

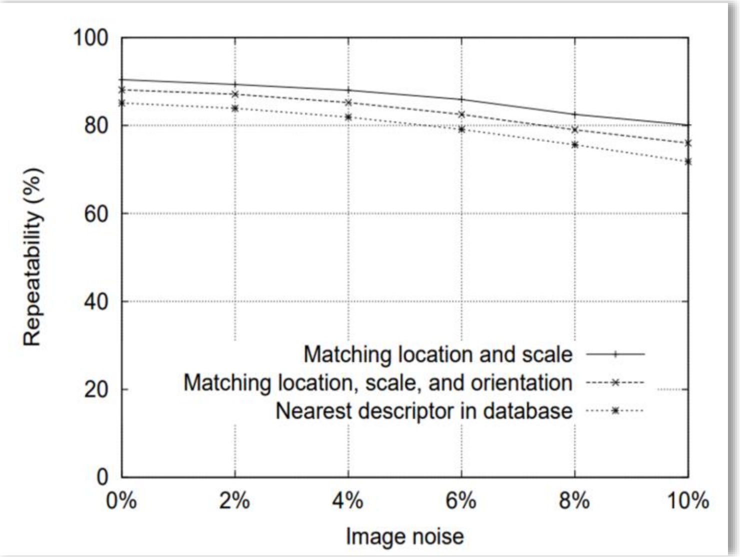
SIFT

3. Orientation assignment

Algorithm

2. 검출된θ의 방향을 **360도 기준으로 10도 씩** 총 36개의 bin을 가지는 **orientation histogram** 생성
3. θ별로 해당하는 **magnitude x (키포인트의 스케일 x 1.5)** 을 통해 얻은 값을 **orientation histogram**에 저장
4. **가장 높은 peak**의 방향으로 **keypoint** 생성
(**가장 높은 peak의 80%** 이상인 다른 로컬 피크가 있으면 해당 방향으로도 **keypoint** 생성)
- 약 15%정도의 포인트가 생성됨(안정성에 중요한 기여)

Image noise에 대한 repeatability



방향 할당이 **15도** 이내여도 **95%의 정확도**로 안정적으로 유지됨
(image noise가 10% 추가된 경우까지도)
-> 노이즈에 강인하고 오류의 주된 원인은 **초기 위치 및 스케일 감지**에 있다.

SIFT

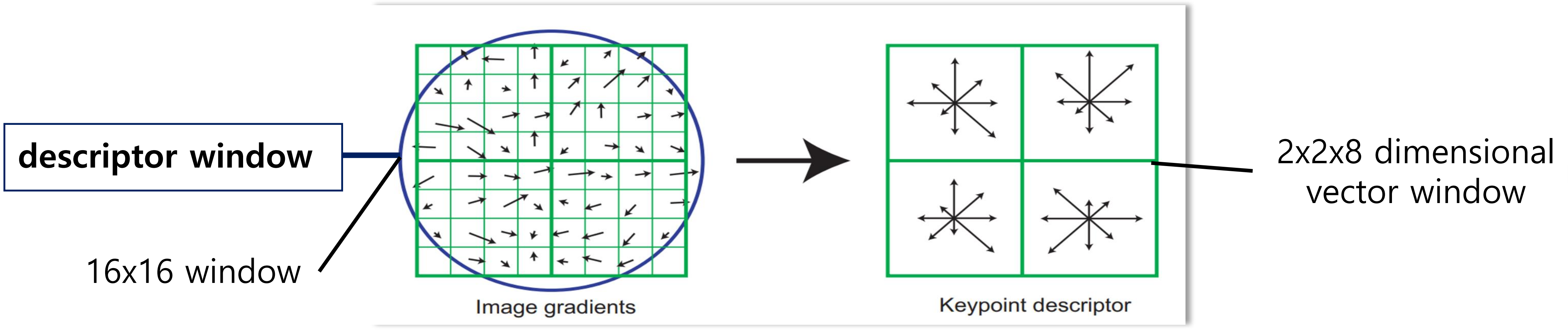
4. Keypoint descriptor

4. The local image descriptor

해당 지역 이미지 영역에 대한 **기술자(descriptor)**를 계산
illumination 이나 **3D viewpoint** 등의 다양한 변형에 가능하게 한 **불변성** 제공

4.1 orientation invariance

1. **descriptor의 좌표**와 **gradient의 방향**은 **keypoint의 방향**을 기준으로 **회전**.
2. **회전된 좌표**와 **방향 정보**를 사용하여 **16x16 window를 4x4 subregion**으로 나눔
3. **각 subregion** 내에서 **gradient magnitudes의 합**을 사용하여 **keypoint descriptor** 생성
4. 각 샘플 포인트 크기에 가중치 할당을 위해 **descriptor window의 너비의 절반의 크기인 σ**를 갖는 **가우시안 가중치 함수** 사용

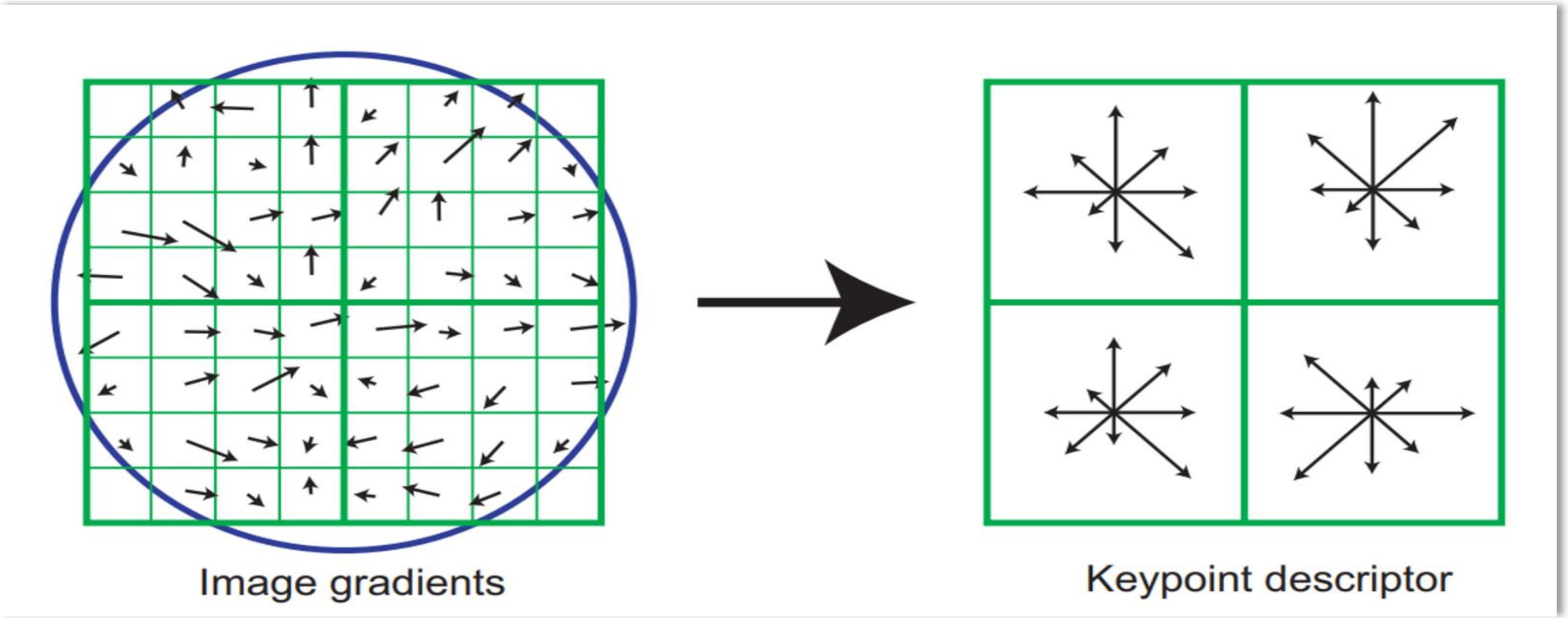


SIFT

4. Keypoint descriptor

4.1 orientation invariance

descriptor가 갑자기 변하는 경우 **boundary affects** 방지책
1. 각 항목(bin)은 각 차원에 대해 (1-d)의 가중을 받는다. (d는 bin의 central value로부터 각 unit까지의 거리)
2. **trilinear interpolation** 사용하여 각 기울기 샘플의 값을 인접한 히스토그램 항목으로 분산 (x,y,d를 사용)



SIFT

4. Keypoint descriptor

4.2 Illumination invariance

Linear illumination change
- 벡터를 단위 길이로 정규화

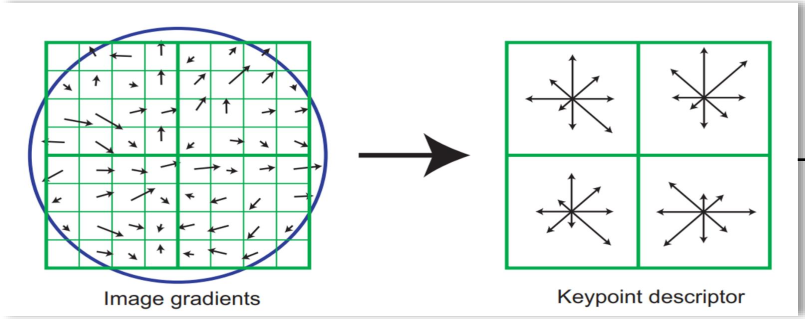
이미지 대비 변경은 각 픽셀 값이 상수로 곱해진다.
➔ Gradient는 픽셀간 차이에서 계산되기 때문에 동일 상수로 곱한다.
➔ 정규화를 하면 위의 이미지 대비 변경에 영향을 받지 않는다.
➔ 즉, 선형 조명 변화에 대해 기술자는 불변

Non-Linear illumination change
- 벡터의 각 값을 0.2보다 크지 않도록 임계 처리 후 단위길이로 정규화

큰 gradient가 특징을 덮어버리는 것을 방지
➔ 비선형 조명변화에 민감하지 않도록 만듦

SIFT

4. Keypoint descriptor

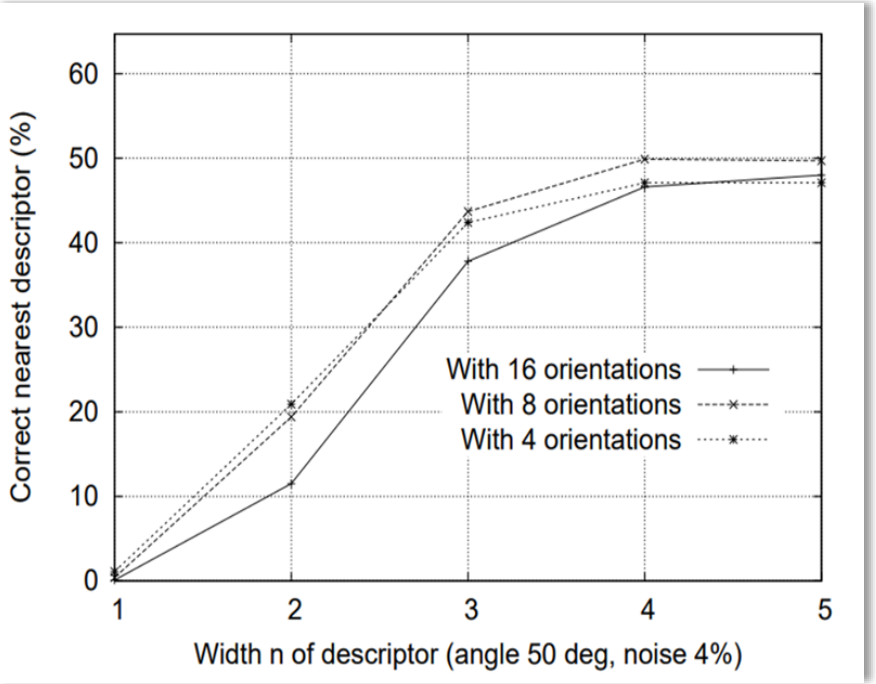


two parameters
- **the number of orientations** r
- **width** n, of the nxn array orientation histograms

복잡성 : rn^2

Trade-off

- 복잡성과 민감성
  - 복잡성이 증가하면 더 나은 구별력을 가짐
  - 복잡성이 증가하면 형태의 왜곡과 가려짐에 더 민감해짐



SIFT

4. Keypoint descriptor


```
keypoint: 1220 descriptor: (1220, 128)
[[ 81.   3.   0. ...   3.   0.   1.]
 [  1.   0.   0. ...   0.   0.   0.]
 [ 32. 100.  53. ...   0.   0. 134.]
 ...
 [ 28.   1.   0. ...  18.  27.  26.]
 [ 17.   6.   0. ...  14. 117.  60.]
 [  4.   0.   0. ...  43.  10.   1.]]
```

감사합니다.