

○ Aliasing 줄이는 방법

Sampling rate ↑

High frequency 제거 → Low-pass filter

낮은 주파수 영역 정보만 통과시키는 필터
대표적인 예: gaussian filter, convolution

○ Cross - Correlation

$$G[i,j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u,v] F[i+u, j+v] \quad G = H \otimes F$$

filter(kernel) image

Convolution 연산이
commutative & associative
고환법칙 결합법칙

○ Convolution

$$G[i,j] = \sum_{u=-k}^k \sum_{v=-k}^k H[u,v] F[i-u, j-v] \quad G = H * F$$

0	0	0
0	0	1
0	0	0

원쪽으로 만큼 shift

1	0	-1
2	0	-2
1	0	-1

X축 방향으로의 차 → vertical(수직) edge

Sobel filter

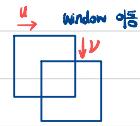
1	2	1
0	0	0
-1	-2	-1

Y축 방향으로의 차 → horizontal(수평) edge

○ Interest Point 인자를 관리하는 방법

↳ key point

○ Harris Corner



설정한 Window를 $[u,v]$ 만큼

shift 했을 때 발생하는 변화량의 합

$$\begin{aligned} E(u,v) &= \sum_{x,y} w(x,y) [I(x+u, y+v) - I(x,y)]^2 \\ &\downarrow \\ I(x+u, y+v) &= I(x,y) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v \\ &= \sum_{x,y} w(x,y) \left[I(x,y) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v - I(x,y) \right]^2 \\ &= \sum_{x,y} w(x,y) \left[\left(\frac{\partial I}{\partial x} \right)^2 u^2 + 2 \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} uv + \left(\frac{\partial I}{\partial y} \right)^2 v^2 \right] \\ &= [u, v] \left(\sum_{x,y} w(x,y) \begin{bmatrix} \left(\frac{\partial I}{\partial x} \right)^2 & \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \\ \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} & \left(\frac{\partial I}{\partial y} \right)^2 \end{bmatrix} \right) \begin{bmatrix} u \\ v \end{bmatrix} \\ &\sim M \end{aligned}$$

$$M = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

공분산 행렬

Window function의 윈도우 영역에서만 1이고 나머지는 모두 0일 때

$$\begin{bmatrix} 0 & 0 \\ 0 & I_y^2 \end{bmatrix} \quad (\because I_x = 0)$$

$$\begin{bmatrix} I_x^2 & 0 \\ 0 & 0 \end{bmatrix} \quad (\because I_y = 0)$$

$$E(u,v) \approx [u \ v] M \begin{bmatrix} u \\ v \end{bmatrix} \rightarrow \text{어떠한 값으로 나를 것인가?}$$

이것을 가지고 corner인지 아닌지 판단

M을 구하기 위해 고윳값 분해 사용

$$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

λ_1 이 크면 λ_2 에 대비되는 eigenvector
방향으로의 변화가 ↑

λ_1 와 λ_2 둘다 크면 corner

λ_1 과 λ_2 중 하나만 크면 edge

λ_1 과 λ_2 둘다 작으면 flat surface

고윳값 계산이 번거로워 실제로 R식을 사용한다!

$$\begin{aligned} R &= \det(M) - k \times (\text{trace}(M))^2 = \lambda_1 \lambda_2 - k \times (\lambda_1 + \lambda_2)^2 \\ &= I_x^2 I_y^2 - (I_x I_y)^2 - k(I_x^2 + I_y^2)^2 \end{aligned}$$

고윳값 모드로
M에 있는 성분 직접 사용

$R > 0$ 이면 corner

$R < 0$ 이면 edge

$|R|$ 이 작으면 flat area

(+) X축과 Y축 방향으로 미분하고,

derivatives 계산하고 gaussian 필터 처리

○ Harris Corner로 Corner 찾기

(1) 여러 개의 Window에 대해 M 행렬 구하기

(2) M을 가지고 R이 threshold를 넘으면 corner의 후보로 찾기

(3) Local maxima 구하기 (Non-maximum Suppression)

이렇게 찾은 corner의 위치는 scale에 invariant하지 X → SIFT

Gaussian kernel of width 0 를 두 번 하면 → Gaussian kernel of width $0\sqrt{2}$

$0 \uparrow$ → 주변부의 정보를 많이 가져온다 → kernel size ↑

○ Gaussian Pyramid

Gaussian filter를 적용하여 이미지 피라미드 구성

Gaussian filter 적용 → sub sampling → 반복... → aliasing 줄이기 위해

$\frac{1}{2}$ 크기로 $\frac{1}{2}$ 크기로 줄여줄 때마다 gaussian filter size는 두 배로

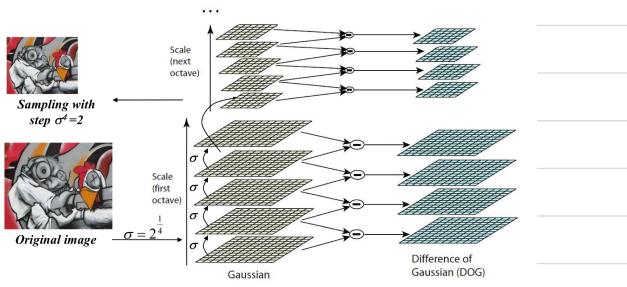
Gaussian filter를 적용하는 게 scale의 관점에서

이미지 size를 줄이는 것과 같은 효과를 준다

- SIFT (Scale Invariant Feature Transformation)
illumination, transition, rotation, scale invariant

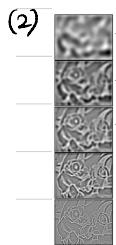
(1) DoG 구하기

1) Gaussian scale pyramid 생성하기



한 octave에는 같은 크기의 이미지에 gaussian 적용하여 scale↑
다음 octave로 넘어갈 때는 scale 유지하면서 subsampling

2) 각 octave마다 gaussian 적용된 이미지 사이의 차이인 DoG 구함



DoG에서 각 이미지마다
인접한 두 DoG 이미지를 포함하여
point마다 이웃한 26개의 pixel 확인
9개 + 8개 + 9개
local maxima인 point → key point

(3) key point마다 주 방향 (dominant orientation) 할당 히스토그램으로 영역 내 픽셀마다 gradient 방향 세었을 때 가장 큰 bin의 방향

(4) 4x4 크기의 16개 (4x4) window 각 window마다 8개의 방향에 관한 count 정보 key point 주 방향으로 normalize 하기

(5) 16개 x 8개 방향 = 128개의 정보 → feature descriptor 생성 밝기 의존성 해결 위해 1로 정규화하고 threshold 적용 후 재정규화

Difference of DoG
scale space에 관해 1차 미분 알고 2차, 3차 미분하는 방법을 사용하면
더 성능 향상 가능

○ Harris Laplace

(1) Gaussian scale pyramid 생성 시 gaussian이 아니라 laplacian of gaussian을 적용

$$\frac{\partial^2 I(x,y)}{\partial x^2} \quad \frac{\partial^2 I(x,y)}{\partial y^2}$$

(2) 적용한 각 scale 이미지에 관해 Harris corner로 local maxima 구하기

(3) 인접한 scale 이미지를 참고하여 scale에 관해 local maxima인 point → key point

$$\left(\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y} \right) / \sigma$$

- Local Descriptor → Robust & distinctive

- SURF (Speeded Up Robust Feature) Hessian 사용
precise localization 가능

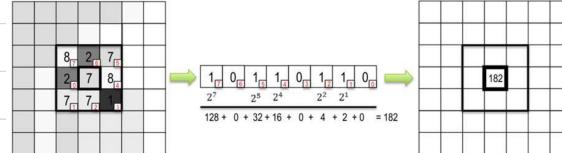
$$\begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix}$$

4x4 크기의 16개 (4x4)의 각 region에 관해

$$V = (\sum dx, \sum dy, \sum |dx|, \sum |dy|)$$

→ 4개의 원소 × 16개 = 64 차원

○ LBP (Local Binary Patterns)



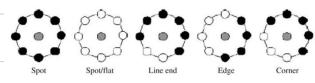
주위 8개의 픽셀에 관해 (반)시계 방향으로 보기

가운데 픽셀보다 값이 크거나 같으면 1 아니면 0 → 8개의 bit를 10진수로
illumination invariant (transition, rotation, scale X)

(+) Uniform LBP : 0번 또는 2번의 bitwise transition 발생 시

$$\frac{P(p-1)}{2} \times 2 + 2 + 1$$

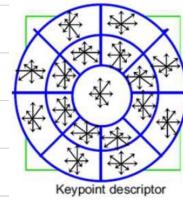
모두 같은 값 memory 절약 가능



Multi-scale LBP : 한 point에 관해 다양한 scale에서 구한 LBP 합치기

○ GLOH (Gradient Location Orientation Histogram)

$$(17\text{개의 log-polar location bin} \times 17 \times 16) / 16\text{개의 orientation bin} = 272 \text{ 차원}$$



128 차원으로 PCA

○ DAISY log-polar grid arrangement 27개의 영역

overlap을 통해 image transformation에 더 강건한 descriptor



○ LATCH (Learned Arrangements of Three Patch Codes)

3개의 key point 주변 픽셀 patch 비교해서 single bit 생성

anchor : $P_{t,a}$ companion : $P_{t,1}, P_{t,2}$

$$f(P_{t,a}, P_{t,1}, P_{t,2}) = 1 \text{ iff } |P_{t,a} - P_{t,1}| < |P_{t,a} - P_{t,2}|$$

DNN은 scale에 관해 파악이 어려워서 아직까지는 SIFT가 많이 사용됨

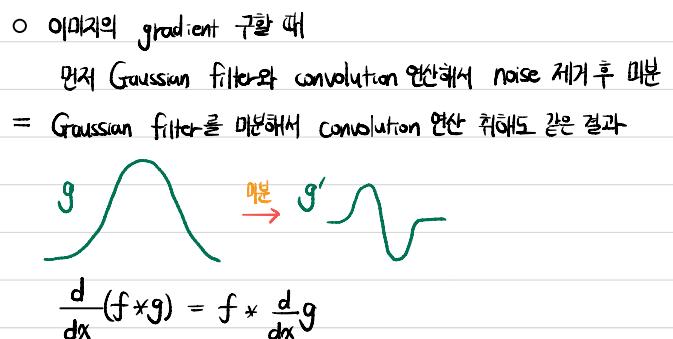
○ LIFT (Learned Invariant Feature Transform)

3개의 CNN 모델 Detector, Orientation Estimator, Descriptor
key point 찾기 → major orientation 파악 ↑

Descriptor 학습 시 loss 함수 [positive pair끼리는 loss 줄이고
negative pair끼리는 loss 늘리고]

○ Filter 종류

Prewitt $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$	\rightarrow x축 방향의 차 → 대각선	$M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$	\rightarrow y축 방향의 차 → 수평선
Sobel $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$	\rightarrow 대각선	$M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$	\rightarrow 대각선
Roberts $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$	\rightarrow 대각선	$M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	\rightarrow 대각선



○ Gradient $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \quad \tan\theta = \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \quad \theta = \tan^{-1}\left(\frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}}\right)$$

○ Template matching 방법 g : filter, f : image

(1) Correlation

$$h[m, n] = \sum_{k, l} (g[k, l])(f[m+k, n+l])$$

(2) Zero-mean Correlation, filter의 mean fast하지만 정확도↓

$$h[m, n] = \sum_{k, l} (g[k, l] - \bar{g})(f[m+k, n+l] - \bar{f})$$

(3) SSD (Sum Square Difference) 부분적으로 이용하지거나 빛나기면

$$h[m, n] = \sum_{k, l} (g[k, l] - f[m+k, n+l])^2 \quad \text{정확도↓}$$

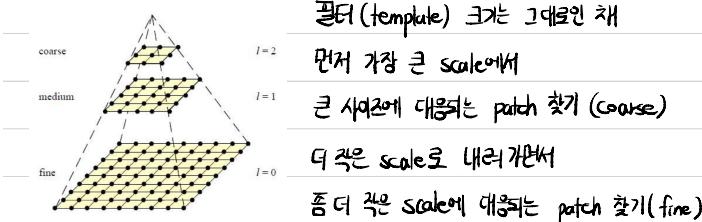
(4) Normalized cross correlation

$$h[m, n] = \frac{\sum_{k, l} (g[k, l] - \bar{g})(f[m+k, n+l] - \bar{f}_{m, n})}{\sum_{k, l} (g[k, l] - \bar{g})^2 (f[m+k, n+l] - \bar{f}_{m, n})^2}$$

노이지로 정확도 best 마치 $\frac{\text{공분산}}{\text{각별의 분산}} = \text{상관계수처럼}$

○ Coarse-to-fine Image Registration

Gaussian pyramid 생성



○ Extrapolate (Padding)

Edge 근처에서 filter로 matching되는 patch 찾기 위해

clip filter: black 등 한 가지 색으로 가득 차도록 채우기

Symmetric: 테두리 기준으로 대칭 복사

replicate: 테두리 값 그대로 복제

circular: wrap around 이미지 주기적으로 반복되어 끝싸는 것

○ Projection matrix

이미지는 실제 3차원 공간상에 있는 점들을 이미지 평면에 투영시킨 것

$$\text{이미지 좌표계} \quad \text{real world 좌표계}$$

$$X = K [R \ t] X$$

Intrinsic parameter

카메라 내부의 모양

Extrinsic parameter

카메라 외부의 모인 (카메라 설치 높이, 방향 등)

$$W \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} fx & skew_c & Cx \\ 0 & fy & Cy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_{11} & h_{12} & h_{13} & t_1 \\ h_{21} & h_{22} & h_{23} & t_2 \\ h_{31} & h_{32} & h_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

rotation \rightarrow translation

fx : focal length가 가로방향 cell 크기의 몇 배인지
 fy : focal length가 세로방향 cell 크기의 몇 배인지

(Cx, Cy) : pinhole(COP, Center of Point)에서 이미지 센서에 내린

수선의 불의 영상좌표 (주점, optical center)

Skew_c: 이미지 센서의 cell array의 y축에 기울어진 정도

↑ x축 방향으로 shearing한 정도

예) intrinsic: 정사각형 이미지 cell, 주점 $(0,0)$, skew x

extrinsic: rotation x, translation x

$$X = K [I \ 0] X$$

$$W \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$wu = fx \quad u = \frac{fx}{Z}$$

$$wv = fy \quad v = \frac{fy}{Z}$$

$$w = Z$$

예) intrinsic: 정사각형 이미지 cell, 주점 (u_0, v_0) , skew x

extrinsic: rotation x, translation x

$$X = K [I \ 0] X$$

$$W \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & u_0 & 0 \\ 0 & f & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$wu = fx + u_0 Z \quad u = \frac{fx + u_0 Z}{Z} = \frac{fx + u_0 Z}{Z}$$

$$wv = fy + v_0 Z \quad v = \frac{fy + v_0 Z}{Z} = \frac{fy + v_0 Z}{Z}$$

$$w = Z$$

예) intrinsic: 직사각형 이미지 cell, 주점 (u_0, v_0) , skew x

extrinsic: rotation x, translation x

$$W \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a & 0 & u_0 & 0 \\ 0 & b & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$W \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a & s & u_0 & 0 \\ 0 & b & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

예) intrinsic: 직사각형 이미지 cell, 주점 (u_0, v_0) , skew x

extrinsic: rotation x, translation (tx, ty, tz)

$$W \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a & 0 & u_0 & tx \\ 0 & b & v_0 & ty \\ 0 & 0 & 1 & tz \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

* Rotation matrix

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

$$R_z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

예) intrinsic: 직사각형 이미지 cell, 주점 (u_0, v_0) , skew x

extrinsic: rotation y $\rightarrow \beta$, translation (tx, ty, tz)

$$W \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a & s & u_0 \\ 0 & b & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta & tx \\ 0 & 1 & 0 & ty \\ -\sin\beta & 0 & \cos\beta & tz \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

최대 degree of freedom [intrinsic: 5

extrinsic: 6

○ Orthographic (Parallel) Projection

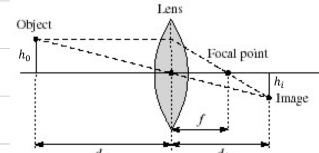
$$W \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

○ 물체와 lens 사이 거리 : d_o

lens와 이미지 사이 거리 : d_i

lens와 focal point 사이 거리 : f

↑ focal length



$$\frac{1}{d_o} + \frac{1}{d_i} = \frac{1}{f}$$

Field of View (회각)

○ focal length $\downarrow \rightarrow$ FoV \uparrow (wide camera) 가정거리 와곡 발생
 focal length $\uparrow \rightarrow$ FoV \downarrow (telescopic)

Depth of Field (얼마나 더 멀리 있는 영역까지 선명히 보이는지)

○ aperture 크기 $\downarrow \rightarrow$ DoF \uparrow (F-number \uparrow)

aperture 크기 $\uparrow \rightarrow$ DoF \downarrow (F-number \downarrow)

○ Chromatic aberration (색수차)

파장에 따른 굴절률의 차이에 의해 생기는 수차

○ Geometrical distortion

주변부에서 왜곡 더 잘 일어남

○ HSV

hue : 색조 예) 빨간색 계열인지 등 (Red: 0°, Green: 120°, Blue: 240°)

saturation : 얼마나 색이 pure한지

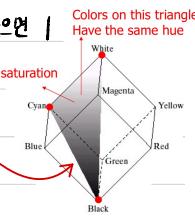
white에 가까우면 0, primary color에 가까우면 1

brightness (value) : 밝기 (intensity) 어두으면 0, 밝으면 1

achromatic notion of intensity

Color 간 decision boundary?

X, Y 평면(비단면)에 perpendicular



○ RGB to HSV

RGB 값을 0에서 1 사이로 정규화

$$R' = R / 255 \quad G' = G / 255 \quad B' = B / 255$$

정규화한 RGB 값에서 최솟값과 최댓값 고르고 그 차이 구하기

$$C_{\max} = \max(R', G', B') \quad \Delta = C_{\max} - C_{\min}$$

$$C_{\min} = \min(R', G', B')$$

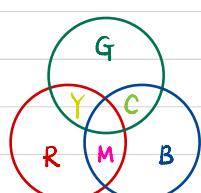
$$H = \begin{cases} 0^\circ & \Delta = 0^\circ \text{ (RGB 값이 모두 같은 경우)} \\ 60^\circ \times \left(\frac{G-B'}{\Delta} \bmod 6 \right) & C_{\max} = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2 \right) & C_{\max} = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4 \right) & C_{\max} = B' \end{cases}$$

$$S = \begin{cases} 0 & C_{\max} = 0 \\ \frac{\Delta}{C_{\max}} & C_{\max} \neq 0 \end{cases} \quad V = C_{\max}$$

서로 대립되는 색

○ Lab L: luminance a: red \leftrightarrow green b: yellow \leftrightarrow blue
perceptually uniform color space

○ RGB와 CMY



$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

빛에서는 second primary color
pigment에서는 primary color

색 모델 변환 (예: RGB \rightarrow HSV)은 색 모델의 viewpoint를 바꾸는 linear transformation으로 ↪

○ 이론적으로 Image processing에서 어떠한 color model을 사용해도 상관 없지만
실질적으로 어떠한 연산을 하느냐에 따라 특별히 더 적합한 모델이 있다.

예) HSV가 간단한 color classification에서는 효과적

(RGB는 color boundary가 perpendicular plane이 되지 못함으로)

per-color component processing

vector processing \rightarrow RGB 값이 서로 연관되어 있을 때 유리함

○ CIE XYZ model

인간의 3색의 원추세포가 각각 인식하는 파장 영역대를 고려하여 만든 색 model

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.431 & 0.342 & 0.178 \\ 0.222 & 0.707 & 0.071 \\ 0.020 & 0.130 & 0.939 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

normalized tri-stimulus values

$$X = \frac{X}{X+Y+Z} \quad Y = \frac{Y}{X+Y+Z} \quad Z = \frac{Z}{X+Y+Z}$$

$X+Y+Z=1$ 이므로 X, Y 만으로도 모든 색 표현 가능

○ 실제 색은 광자마다의 광자 양인 스펙트럼

$$\begin{array}{lll} \text{hue} : \text{mean} & \text{saturation} : \text{variance} & \text{brightness} : \text{area} \\ \downarrow & \uparrow & \uparrow \\ \end{array}$$

○ Cones (원추세포) : 색 인식, less sensitive, 주로 fovea(중심)에 많이 분포

Rods (간상세포) : 명암 인식 (gray-scale intensity), highly sensitive

○ Edge 발생 원인

surface normal : 면이 깎아 부분 surface color : 색상 변화

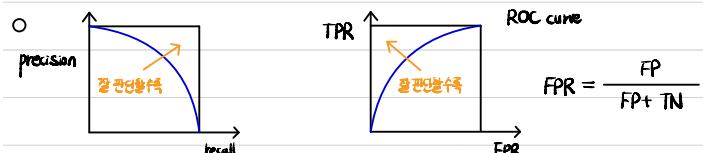
depth : 심도 차이 (서로 다른 거리) illumination : 빛기 변화 (그림자)

$$\text{Recall} = \frac{TP}{TP+FN} \quad \text{Precision} = \frac{TP}{TP+FP}$$

실제로 true인 것 중에서
positive로 예측한 것의 비율

Positive로 예측한 것 중에서
실제로 true인 것의 비율

$$\text{F1-score} = \frac{2 \cdot \text{recall} \cdot \text{precision}}{\text{recall} + \text{precision}}$$



○ Gaussian 0↑ (filter size ↑) \rightarrow smoothing ↑ noise 제거 ↑ localization ↓

○ Non-maximum suppression : locally 가장 큰 gradient 크기를 보이는 부분만 남기고 나머지는 다 interpolation하거나 없애는 것

○ Canny edge detector

1. Gaussian의 x 방향과 y 방향으로 각각 미분한 필터를 이미지에 적용한다

2. Gradient의 크기와 방향을 구한다 \rightarrow edge 방향은 gradient의 수직

3. Non-maximum suppression

4. Hysteresis thresholding

Strong edge 연결되어 있는
Weak edge를 edge로 linking
edge x
high threshold
low threshold
 아닌 건 모두 제외

- Edge detection → 이미지 픽셀이 edge인지 아닌지 구분하는
binary classification

DNN은 아래 같은 binary classification에 유리함

- 사람마다 edge detection 결과 다를 수 있음