9. 컴퓨터 비전

1. 컴퓨터 비전의 문제
2. 영상의 표현
3. 영상처리
4. 특징추출
5. 컴퓨터 비전의 대상
6. 객체 위치 검출 및 개체 인식
7. 의미적 영역 분할
8. 딥러닝 응용

1. 컴퓨터 비전

- **컴퓨터 비전**(computer vision)
  - **2차원 영상**으로부터 **3차원 장면**을 **재구성**하여, **해석**하고, **이해**하는 것을 연구하는 분야
  - 궁극적인 목적
    - 컴퓨터 소프트웨어와 하드웨어를 사용하여 **인간의 시각**을 **모델링**하여 **복제**해 내는 것
  - 인간 시각에 비하면 아직 **전체적으로 초보적인 수준**
    - **얼굴인식, 물체 감지** 등의 특정 문제에서는 인간 이상의 성능 달성

컴퓨터 비전

❖ 컴퓨터 비전의 본질적 어려움

- **역문제**(inverse problem)
  - 2차원 영상에서 3차원 정보 재구성 필요

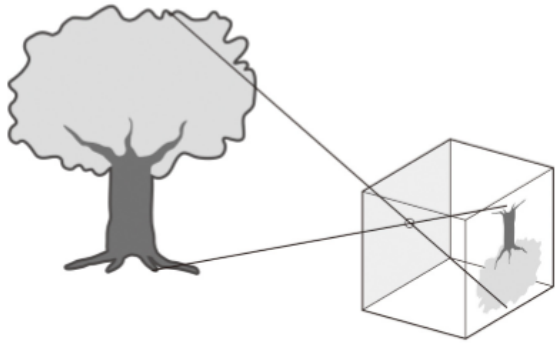


그림 9.1 핀홀(pinhole) 카메라를 통한 영상 획득

- **불량 문제**(ill-posed problem)
  - 대부분의 컴퓨터 비전 문제는 답이 유일하지 않음

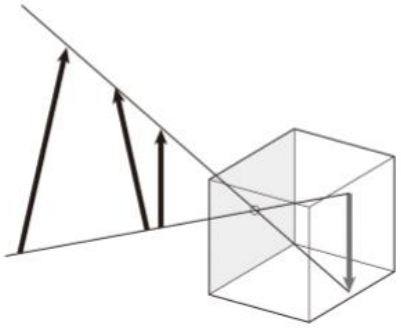


그림 9.2 다대일 투영관계로 인한 컴퓨터 비전의 불량 문제

- **영상 생성과정**에 **기하학적 변환**(회전, 크기 변경, 투영 등), **광도 변환**(조명, 그림자 등), **광학적 잡음** 발생

컴퓨터 비전

❖ **컴퓨터 비전 적용 사례**
- 손글씨로 쓴 우편번호 인식 및 자동차 번호판 **인식**
- 품질 보증을 위한 **기계 검사**
- 사진을 보고 **3차원 모델**을 만드는 **기술**
- 항공 사진을 이용한 **3차원 모델**의 자동 **생성**
- 의료 **영상 판독**
- 보행자 보호 및 안전 운전을 위한 자동차 **안전 보조 장치**
- **모션 캡처**
- **경계** 및 **감시**
- 지문 등 **생체 신호분석**
- 여러 사진을 자연스럽게 이어 붙이는 **파노라마 사진 제작**
- 여러 영상을 **합성**하는 방법
- **3차원 모델링**
- 형태를 점진적으로 바꿔가는 **모핑**(morphing)
- **얼굴 인식**
- 시각적 **인증**
- …

1.1 컴퓨터 비전의 관련 분야

❖ **컴퓨터 비전 관련 분야**
- 영상처리, 패턴인식, 기계학습, 컴퓨터 그래픽스 등

❖ **영상처리**(image processing)
- 입력 받은 영상을 사용 목적에 맞게 적절하게 처리하여 보다 **개선된 영상**을 생성하는 것
- 입력 영상에 있는 **잡음(noise) 제거**, 영상의 **대비**(contrast) **개선**, **관심영역**(region of interest) 강조, **영역 분할**(segmentation), **압축** 및 **저장** 등
- 영상처리와 컴퓨터 비전은 많은 부분 중첩
- **영상처리**는 주로 전처리 등 **저수준 처리**에 활용,
  컴퓨터 비전은 **고수준 처리**

1.1 컴퓨터 비전의 관련 분야

❖ **패턴인식**(pattern recognition)

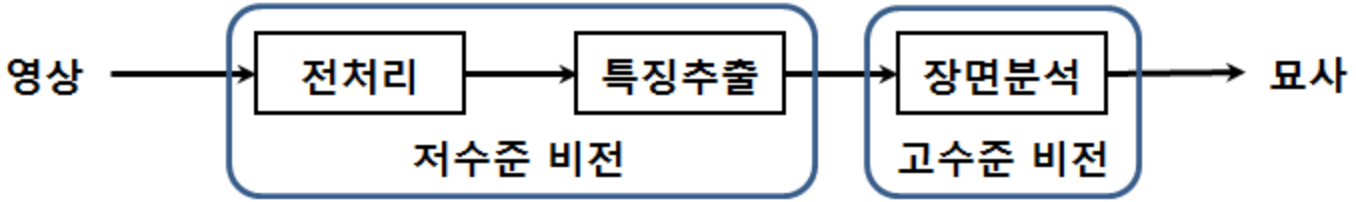
- 다양한 데이터에 대해서 **패턴**을 **추출**하고 이를 이용하여 입력 데이터를 **특정 부류**(class)로 **분류**하는 방법
- 영상,비즈니스 데이터, 음성 신호, 과학실험 데이터 등 다양한 데이터에 관심
- 다양한 **기계학습 기술** 활용
- 컴퓨터 비전에서 **개체 식별**, **범주 인식** 등 인식에 관련된 부분에 사용

❖ **컴퓨터 그래픽스**(computer graphics)

- 데이터 형태로 **3차원 모델**을 만들고 3차원 모델을 사용하여 **2차원 영상 생성**
- 영상으로부터 3차원 모델을 생성하고, 이를 바탕으로 컴퓨터 그래픽스 기술을 이용하여 원래 영상에 대응하는 영상을 만들어 **합성**하는 기술

1.2 컴퓨터 비전의 처리 단계

❖ 컴퓨터 비전의 **처리 단계**



- **전처리 단계**
  - 주로 영상처리 기술 사용
  - 다양한 **특징 추출**
    - 에지(edge), 선분, 영역, SIFT(Scale-Invariant Feature Transform) 등

- **고수준 처리**
  - 특징정보를 사용하여 영상을 **해석**, **분류**, **상황묘사** 등 정보 생성

2. 영상 표현

❖ **화소(pixel, 畵素)**

- 디지털 영상을 표현하는 2차원 배열에서 각 원소
- 해당 위치에서 빛의 세기에 대응하는 값
  - 0은 검은색을 나타내고, 화소값이 커질수록 밝은 색

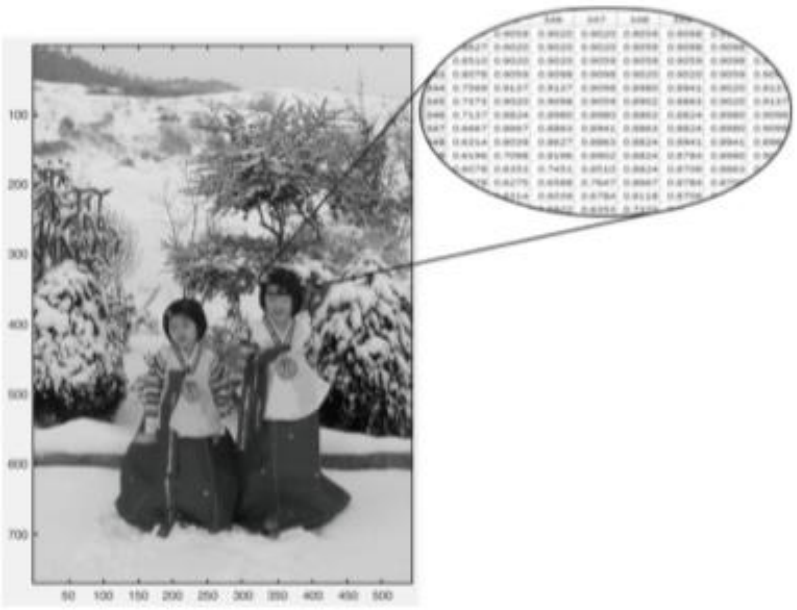


그림 9.4 디지털 영상의 표현

- **칼러 영상**
  - **R**(red), **G**(green), **B**(blue) 세 가지 색상에 대한 정보 화소 정보 표현
  - 2차원 행렬 3개로 표현

영상 표현

❖ 영상의 좌표계

- 첨자(index)
  - 원점 (0,) : 좌측상단 끝
  - (j, i) : 행번호는 y축 첨자, 열번호는 x축의 첨자 사용
  - 첨자는 대개 0부터 시작
- 영상의 **해상도**(resolution)
  - 행렬의 크기

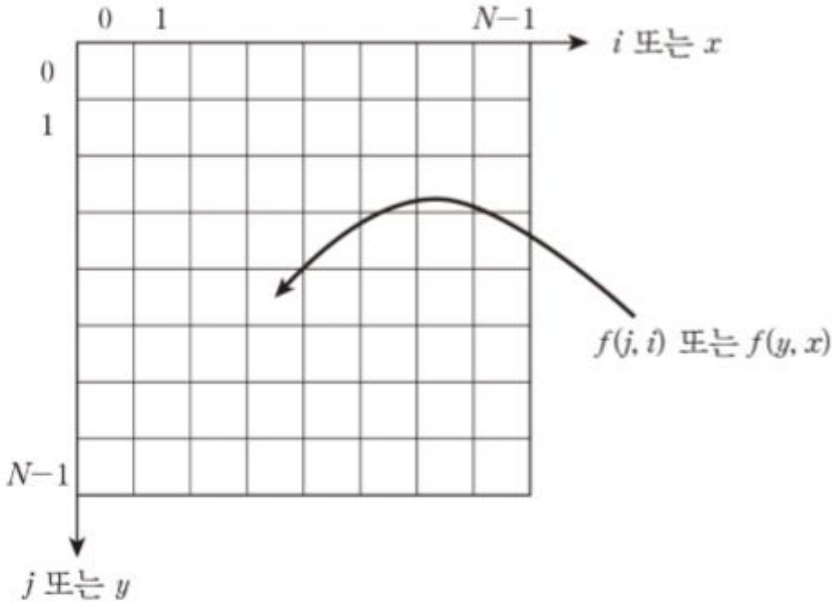


그림 9.5 영상의 좌표계

영상 표현

- **이진영상**(binary image)
  - 화소값이 0 또는 1
- **그레이 영상**(grey image, 명암영상)
  - 일정 범위의 화소값
- **칼러 영상**(color image)
  - 각 칼러 채널별로 그레이 영상과 같이 일정범위의 값
- **압축 저장**
  - 데이터량이 크기 때문에 일반적으로 압축해서 저장
  - 압축을 해제한 상태에서 화소의 값을 접근하여 영상처리 작업
  - 다양한 표준
    - jpeg, mpeg 등

3. 영상 처리

❖ **영상처리(image processing)**
- 영상을 입력으로 받아 영상의 화소값을 변환하는 작업을 수행하여 새로운 영상 출력

❖ **점 연산**(point operation)
- 현재 화소값을 기준으로 출력 영상에 해당 위치의 화소값 결정
- **이진화**, 히스토그램 **평활화**, 여러 영상의 **평균**, 영상 간의 **차연산**, **디졸브** 연산 등

❖ **영역 연산**(area operation)
- 주변 화소의 값을 참고하여 화소의 값 변경
- **컨볼루션**(convolution, **회선**)을 이용한 연산

❖ **기하 연산**(geometric operation)
- 기하학적 규칙에 따라 멀리 떨어진 화소의 값을 참고하여 값 변경
- 영상의 **회전**, **이동**, **확대** 등 **어파인 변환**(affine transformation)

3.1 이진화

❖ **이진화**(binarization)

- 그레이 영상을 이진 영상으로 바꾸는 것

$$f_o(j,i) = \begin{cases} 1 & \text{if } f(j,i) \geq \theta \\ 0 & \text{if } f(j,i) < \theta \end{cases}$$

$f_o(j,i)$:변환후의 화소값
$f(j,i)$: 원본 영상의 화소값

(a) (b)

그림 9.6 영상의 이진화
(a) 원본 그레이 영상 (b) 이진 영상

- 적합한 임계값을 정하는 것이 핵심

3.1 이진화

- **이진화**(binarization)
  - **오츄의 알고리즘**(Otsu's algorithm)
    - 임계값을 효과적으로 결정하는 방법
    - 임계값보다 큰 화소값들의 집단과 임계값보다 작은 화소값들의 집단에 대해서, 두 집단의 분산이 최소가 되도록 하는 임계값 탐색

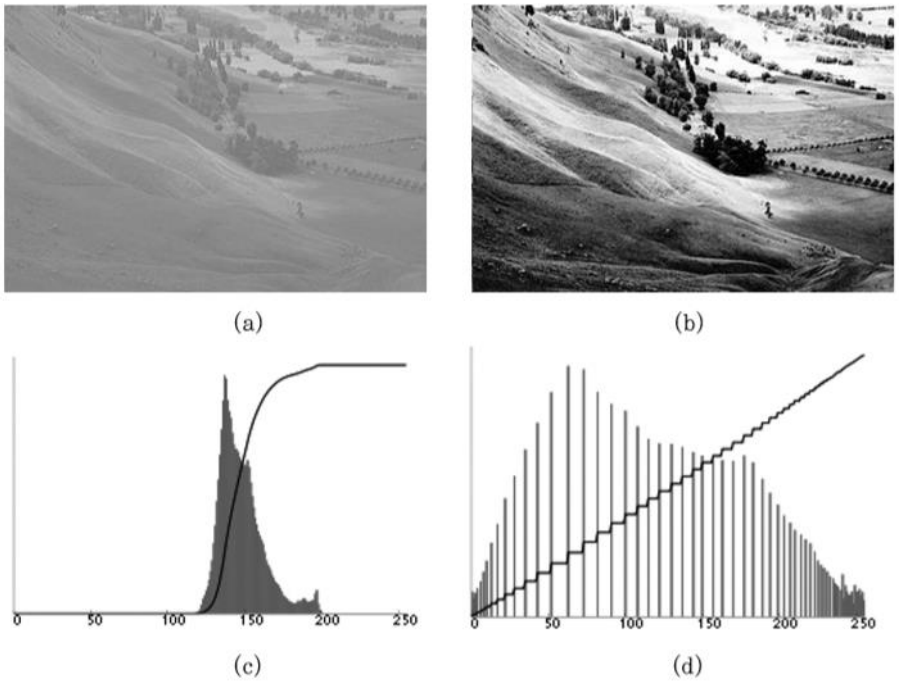
3.2 히스토그램 평활화

❖ **히스토그램 평활화**(histogram equalization)

- 영상의 히스토그램이 전체 영역에 균등하게 분포하도록 변환하여 영상의 대비가 커지도록 해 영상을 선명하게 하는 것

명암값 i까지 누적 빈도값

$$acc[i] = \sum_{j=0}^{i} hist[j] \quad i = 0, \cdots, L-1$$



$$n[i] = acc[i] / N \times (L-1)$$

그림 9.7 히스토그램 평활화

(a) 원본 영상 (b) 평활화 적용 결과 (c) 원본 영상의 히스토그램과 누적분포 곡선 (d) 평활화된 결과에 대한 히스토그램과 누적분포 곡선

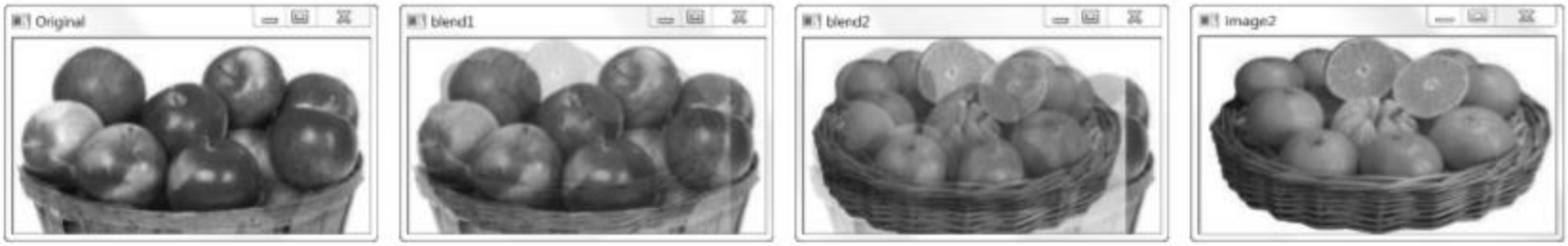
3.3 장면 디졸브

❖ **장면 디졸브**(scene dissolve)

- 두 개의 장면을 **합성**하는 연산

$$f_o(j,i) = \alpha f_1(j,i) + (1-\alpha) f_2(j,i)$$

- α : 앞 장면이 반영되는 비율인데, 1에서 시작하여 0으로 변화



(a) (b) (c) (d)

그림 9.8 장면 디졸브

(a)는 처음 장면이고 (d)는 마지막 장면이다. (b)와 (c)는 디졸브가 진행되어 가면서 (d)의 장면이 점차 나타나는 것을 보여준다.

3.4 컨볼루션 연산과 필터

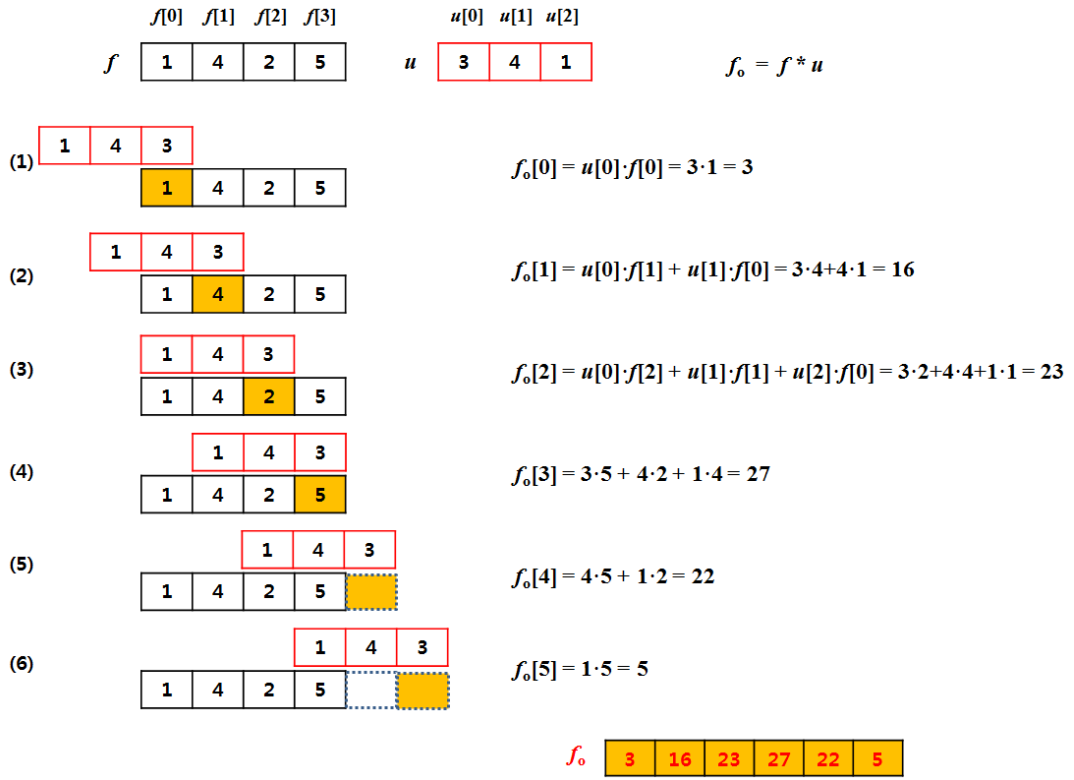
❖ **컨볼루션**(convolution, 회선)

$$f_o[i] = u[0] \cdot f[i] + u[1] \cdot f[i-1] + \cdots + u[k-1] \cdot f[i-k+1]$$

$$= \sum_{j=0}^{k-1} u[j] \cdot f[i-j]$$

$f_o = f * u$

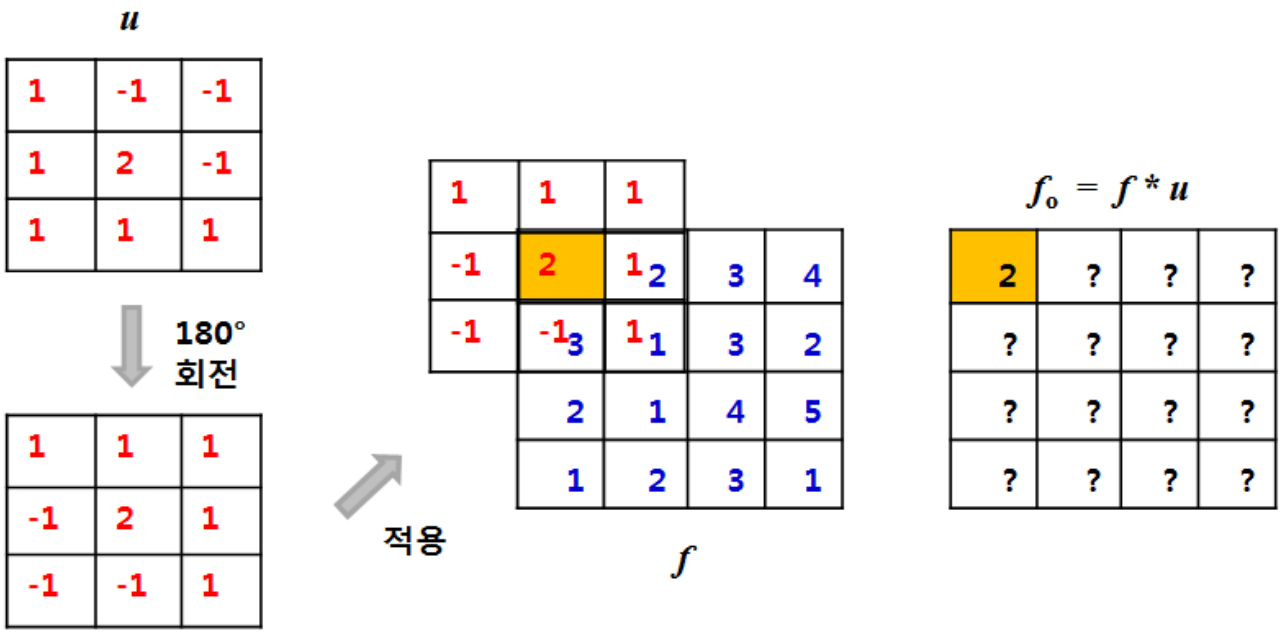
입력 윈도우



컨볼루션 연산과 필터

❖ 2차원 영상에 대한 **컨볼루션 연산**

$$f_o(j,i) = \sum_{y=-(k-1)/2}^{(k-1)/2} \sum_{x=-(k-1)/2}^{(k-1)/2} u(y,x) f(j-y, i-x)$$



- **윈도우** : **마스크**(mask), **커널**(kernel), **필터**(filter), **템플릿**(template)

컨볼루션 연산과 필터

❖ **에지**(edge)

- 영상의 명암, 칼러, 또는 테스쳐가 급격하게 변하는 위치로서 경계선이 나타나는 부분

❖ **Sobel 연산자**

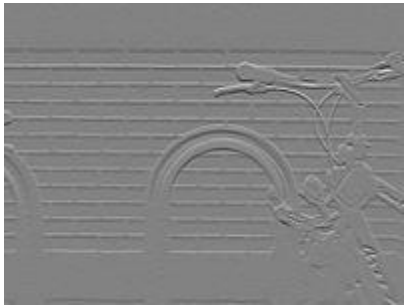
- 에지 검출

$f_1(j,i)$

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

m_x

$f_2(j,i)$



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

m_y

$\sqrt{f_1^2(j,i)+f_2^2(j,i)}$

컨볼루션 연산과 필터

❖ **Prewitt 연산자**(Prewitt operator)

- 에지 검출

-1	0	1
-1	0	1
-1	0	1

m_x

-1	-1	-1
0	0	0
1	1	1

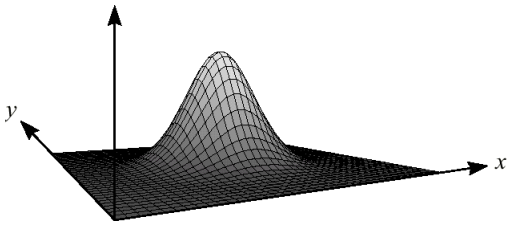
m_y

컨볼루션 연산과 필터

❖ **가우시안 필터**(Gaussian filter)

- 영상에 있는 잡음(noise) 제거
- **가우시안 함수**(Gaussian function)

$$G(y,x,\sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-x^2+y^2}{2\sigma^2}}$$



$\frac{1}{273}$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1

3×3필터 5×5필터 7×7필터

3.5 에지 검출 Canny 연산자

❖ **Canny 연산자**(Canny operator)

- 가장 널리 사용되는 에지 검출 방법
- 과정

1. 영상에 주어진 크기의 **가우시안 필터**를 적용한다.
2. **Sobel 연산자**를 적용하여, **에지**의 **강도**와 **방향**을 구한다.
3. 자신의 **이웃보다 작은 강도**를 갖는 **화소**들을 **제거**한다.
4. 남은 화소들 중에서 **임계값 이상**의 값을 갖는 **화소**에서 시작하여 연결된 **이웃**들을 찾아 **연결**하여 에지를 구성한다.

(a) (b)

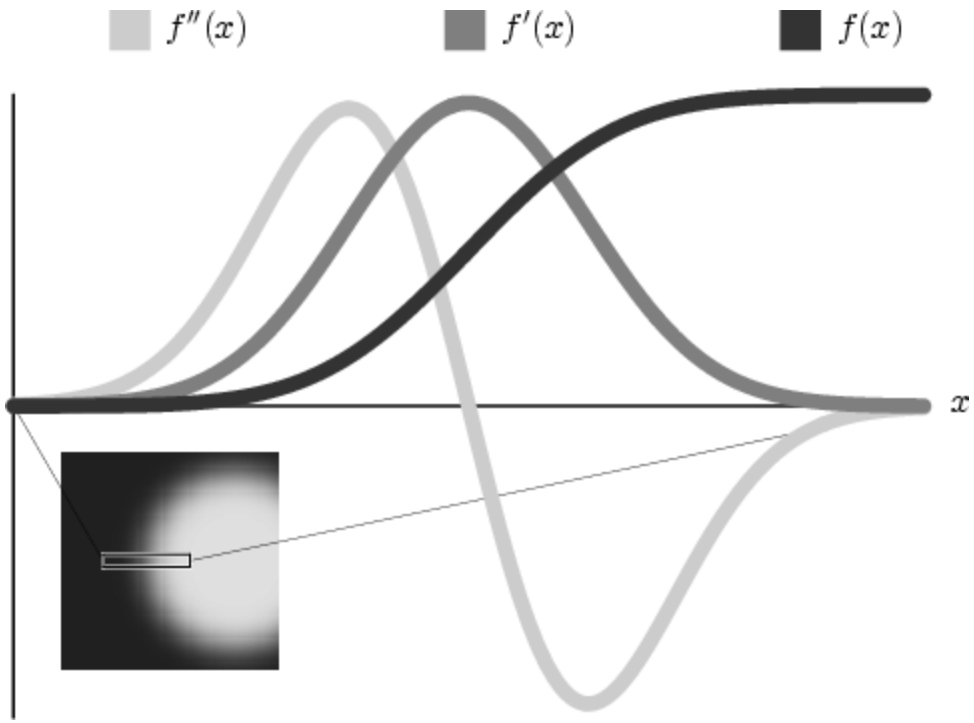
그림 9.17 Canny 연산자의 에지 검출

(a) 원 영상 (b) Canny 연산자 적용 결과

3.6 LOG 필터

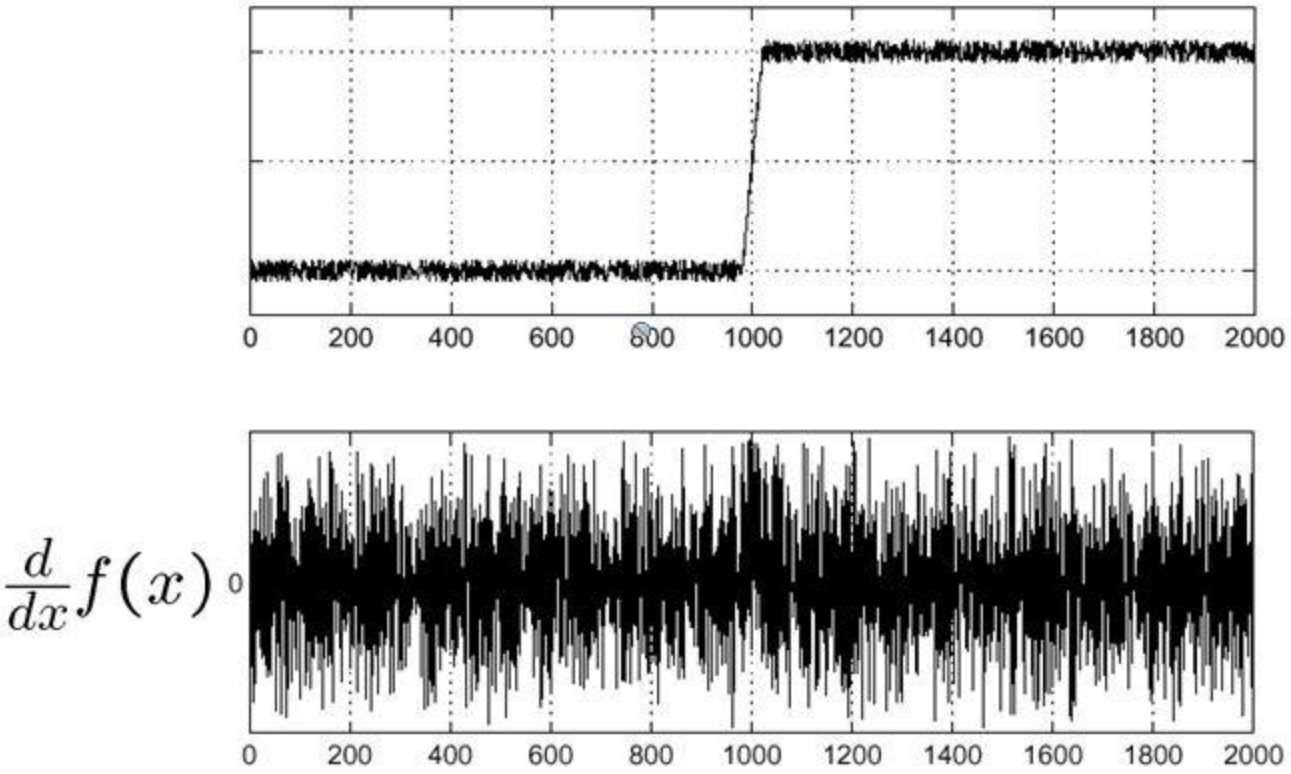
❖ 영상의 1차 미분과 2차 미분

- 에지(edge, 경계선)의 위치



LOG 필터

❖ 오차 신호에 대한 미분



LOG 필터

❖ LOG(Laplacian of Gaussian) 필터

- 영상에 **가우시안 필터**를 적용한 다음, **라플라시안 연산자**(Laplacian operator)를 적용하는 것
- **함수 $f(y,x)$의 라플라시안 $\nabla^2 f$**
  - 함수 $f(y,x)$에 대한 x와 y에 대한 2차 편도함수의 합

$$\nabla^2 f(y,x) = \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial x^2}$$

- **라플라시안 필터(Laplacian Filter)**

$$\begin{aligned}\nabla^2 f(y,x) &= \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial x^2} \\ &= f(y+1,x)+f(y-1,x)-2f(y,x)+f(y,x+1)+f(y,x-1)-2f(y,x) \\ &= f(y+1,x)+f(y-1,x)+f(y,x+1)+f(y,x-1)-4f(y,x)\end{aligned}$$

0	1	0
1	-4	1
0	1	0

LOG 필터

❖ LOG(Laplacian of Gaussian) 필터

- 영상 $f(y,x)$에 가우시안 필터 $G(y,x)$와 라플라스 필터 $L(y,x)$를 적용

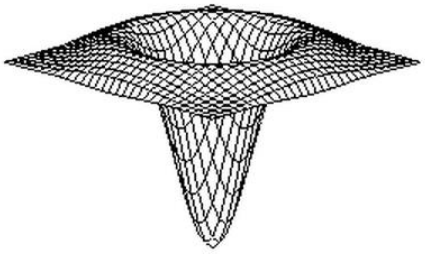
$$LOG(y,x) = L(y,x)*(G(y,x)*f(y,x))$$

- 2번의 컨볼루션 연산에 따른 시간 비용
- 컨볼루션은 **결합법칙** 성립

$$LOG(y,x) = (L(y,x)*G(y,x))*f(y,x)$$

- **가우시안 함수에 대한 라플라시안** $\nabla^2 G(y,x) = L(y,x)*G(y,x)$

$$\nabla^2 G(y,x) = \left(\frac{y^2+x^2-2\sigma^2}{\sigma^2}\right) G(y,x)$$



멕시코 모자 함수
(Mexican hat function)

0.4038	0.8021	0.4038
0.8021	−4.8233	0.8021
0.4038	0.8021	0.4038

3×3필터

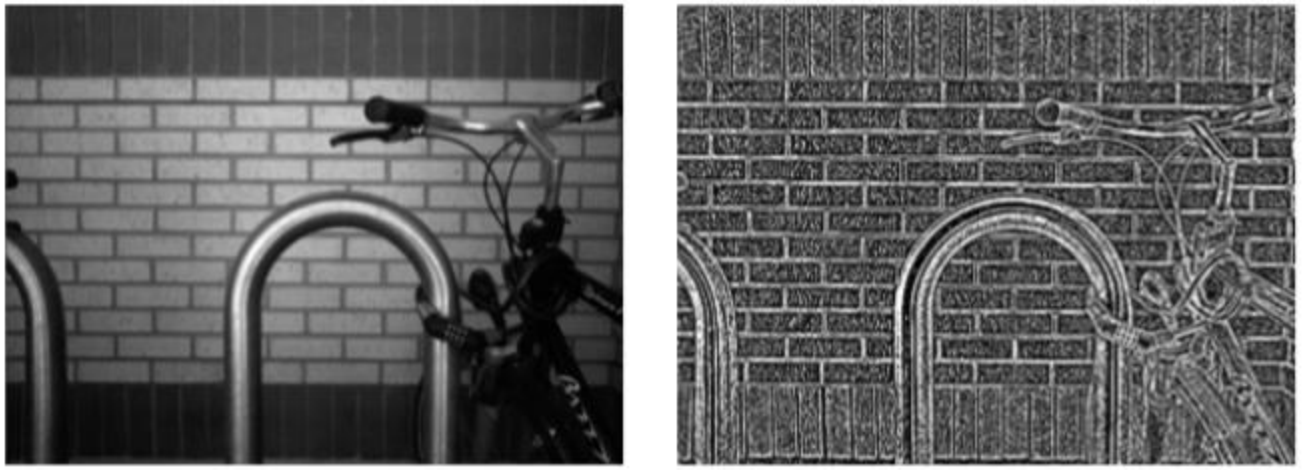
0.0239	0.0460	0.0499	0.0460	0.0239
0.0460	0.0061	−0.0923	0.0061	0.0460
0.0499	−0.0923	−0.3182	−0.0923	0.0499
0.0460	0.0061	−0.0923	0.0061	0.0460
0.0239	0.0460	0.0499	0.0460	0.0239

5×5필터

LOG 필터

❖ LOG 필터 적용 결과

- 에지에 해당되는 부분에서 **영교차**(zero crossing, 양수에서 음수, 또는 음수에서 양수로 변화) 발생
- LOG 필터 적용 결과에 대해서 **영교차 위치**를 찾으면 **에지 검출**



(a) (b)

그림 9.21 LOG 필터의 적용

(a) 원본 영상 (b) LOG 적용 결과

3.7 DOG 연산

❖ DOG(Difference of Gaussian) 연산

- 서로 다른 σ값을 갖는 가우시안 필터를 적용한 두 영상의 차이를 구하는 것

$$DOG(\sigma) = G(k\sigma)*f - G(\sigma)*f = (G(k\sigma) - G(\sigma))*f$$

- **LOG과 DOG의 관계**

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

  - **LOG대신에 DOG 사용 가능**

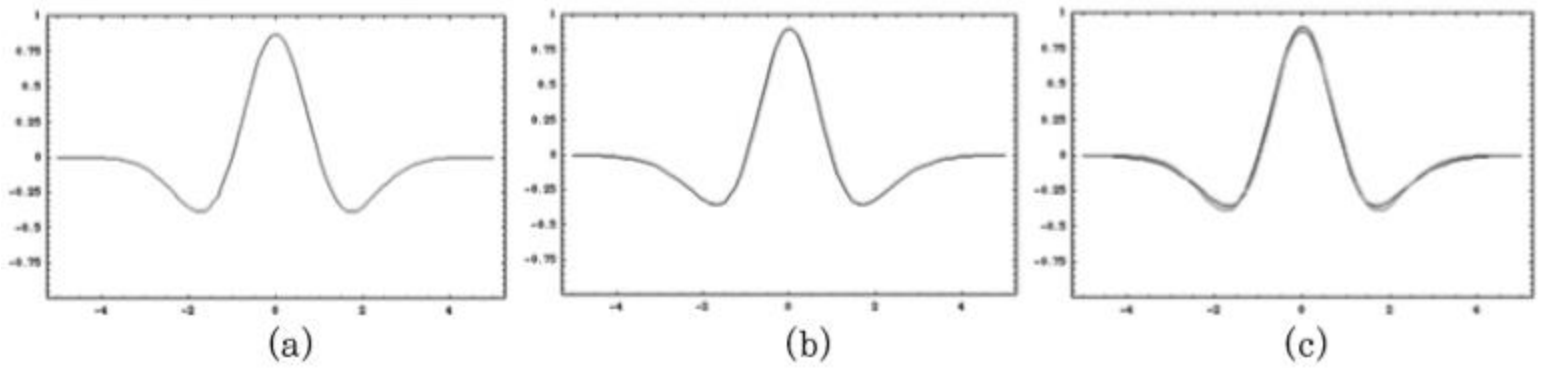


그림 9.22 1차원 가우시안 함수에 대한 LOG와 DOG 함수 형태

(a) LOG (b) DOG (c) LOG와 DOG를 중첩한 그래프

DOG 연산

❖ DOG 연산

- **영교차를 사용하는 에지 검출**

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

  - $(k-1)\sigma^2$ 항은 무시하고 $G(k\sigma) - G(\sigma)$에서 영교차 탐색

- **일정 주파수 범위의 특징 추출**
  - **가우시안 필터**는 신호에서 고주파 성분을 제거하는 효과
  - 서로 다른 σ 값의 가우시안 필터를 적용한 영상의 차(difference)
  - **블롭**(blob, 주변과 구별되는 밝기, 색상과 같은 성질을 갖는 영역) 검출에 사용

DOG 연산

❖ DOG 연산 적용 결과

(a) (b)

그림 9.23 DOG 연산 (a) 원본 영상 (b) DOG 적용 결과

3.8 영상 분할

❖ **영상 분할**(image segmentation)

- 영상을 겹치지 않으면서 전체 영상을 덮는 영역들을 찾아내는데, 각 영역이 유사한 특징을 갖도록 하는 것
- 물체 추적, 영상 검색, 동작 인식, 얼굴 인식 등의 전처리에 필요

(a) (b)

그림 9.24 영상 분할

(a) 원본 영상 (b) 영상 분할 결과

영상 분할

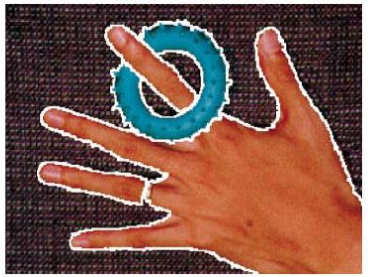
❖ 영상 분할 방법

- **분할후 합병(split-and-merge)** 방법
  - 영상을 4등분하는 일을 분할된 조작이 비슷해질 때까지 계속하다가 더 이상 분할이 되지 않으면 유사한 인접 조건을 결합

- **k-means 알고리즘**을 적용하는 군집화 방법

- **민쉬프트(mean-shift) 군집화** 알고리즘

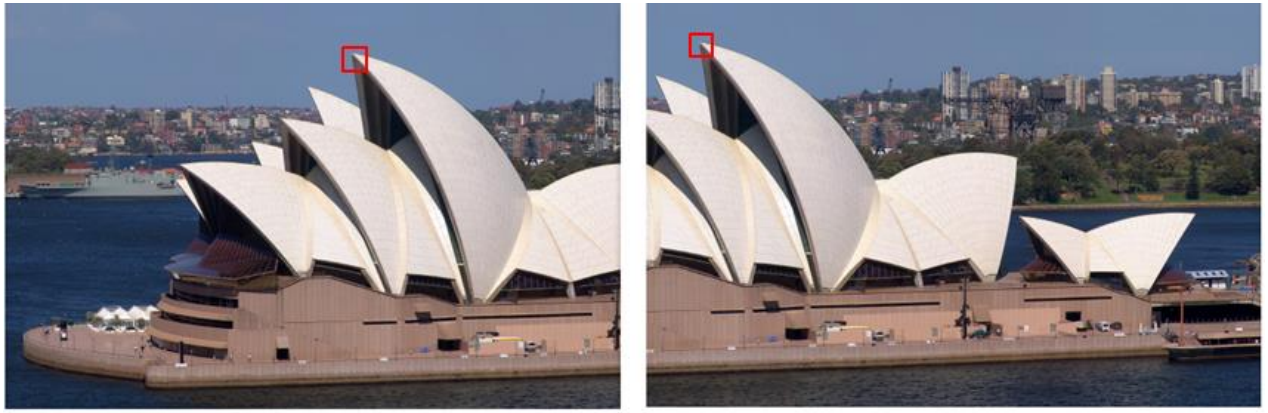
- **그래프 절단(graph-cut) 문제**로 푸는 그래프 기반 알고리즘
  - 영상의 화소를 노드로 간주하고, 이웃의 유사한 것들 사이에는 연결선을 부여하여, 영상을 그래프로 표현한 다음에, 그래프 절단 문제로 해결



4. 특징 추출

❖ 대응점(correspondence point) 찾기 문제

- 두 영상에서 서로 **일치하는 점**들의 **쌍**을 찾는 문제

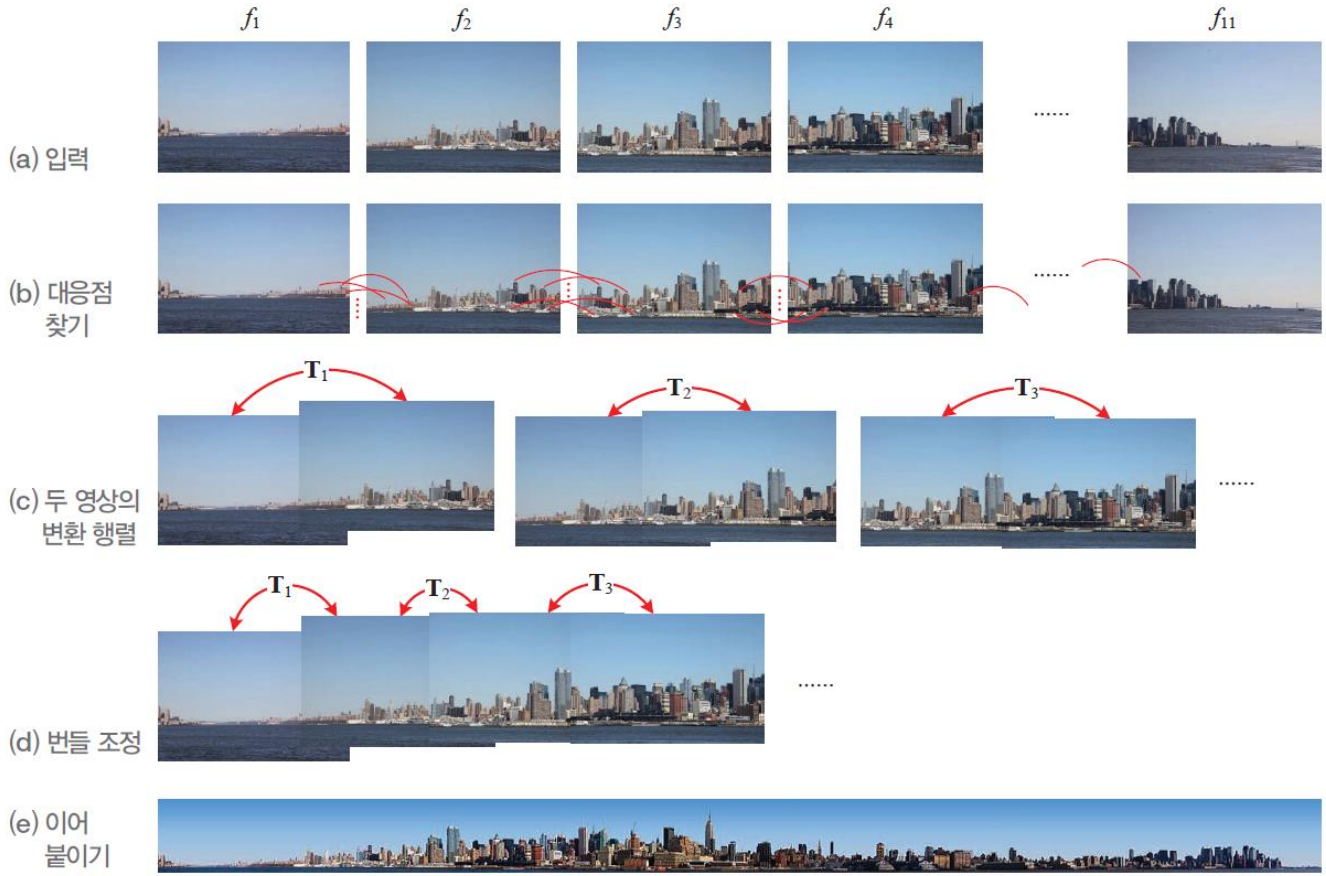


- 영상에서 **특징점**들을 **검출**하고, 특징점들의 **주변정보**를 참고하여 **속성**을 **기술**한 다음, **기술된 정보**를 **비교**하여 일치하는 것을 찾는 과정을 통해 해결
- **파노라마 영상 제작, 물체 인식, 물체 추적, 스테레오 비전, 영상 정합**의 문제 해결에서 필요

특징 추출

❖ 대응점의 활용

- **파노라마 영상**(panorama image)
  - 중첩해서 찍은 인접한 두 사진에서 대응점을 찾아서 두 사진의 대응점들이 겹치도록 합성

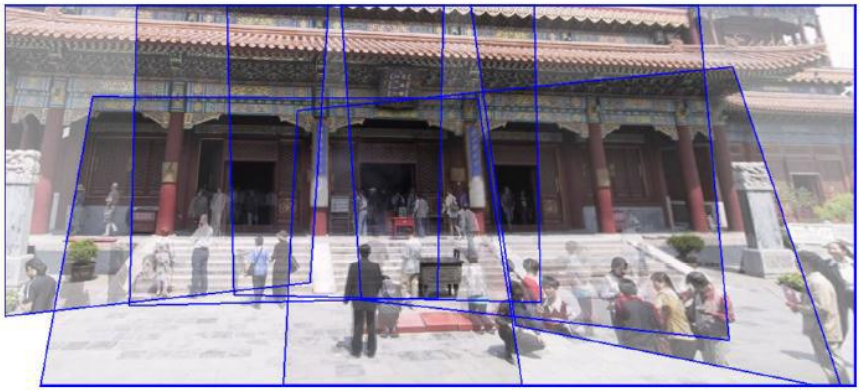


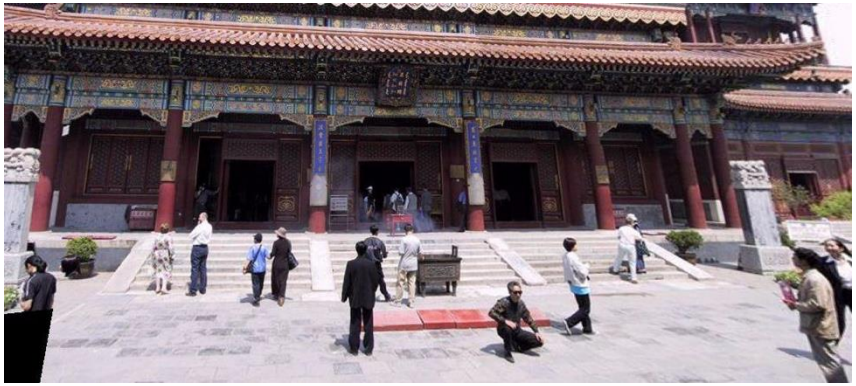
오일석, 컴퓨터비전

특징 추출

❖ **대응점의 활용**

- **파노라마 이어붙이기(panoramic stitching)**
  - 중첩해서 찍은 인접한 두 사진에서 대응점을 찾아서 두 사진의 대응점들이 겹치도록 합성

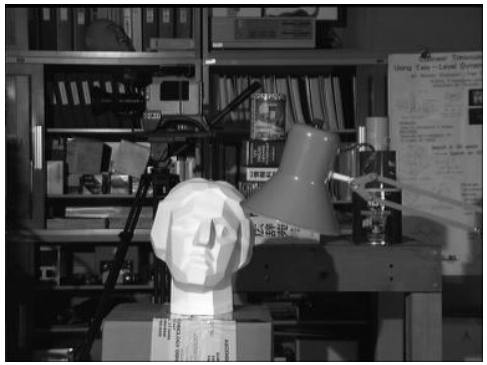




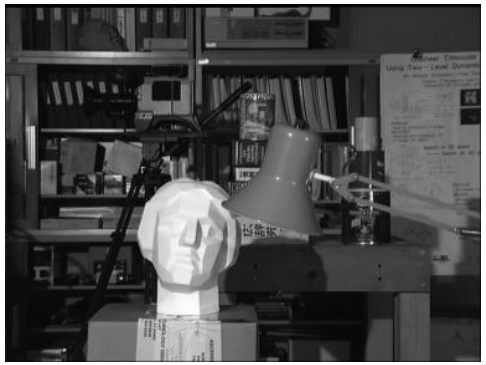
특징 추출

❖ 대응점의 활용

- **스테레오 비전**(stereo vision)
  - 같은 장면을 다른 각도에서 찍은 두 영상으로부터 물체까지의 거리 계산
  - 삼각형의 닮은비로 계산

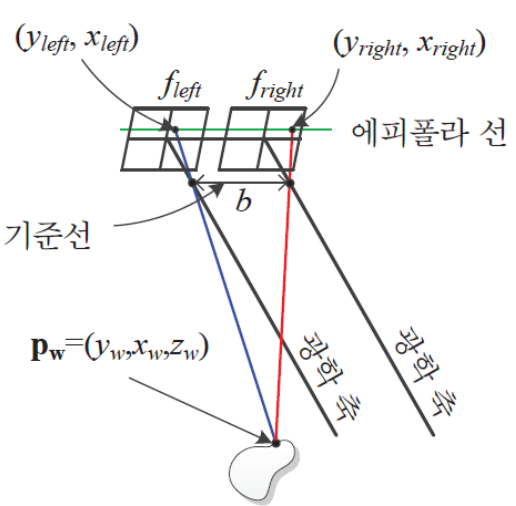


좌측 영상

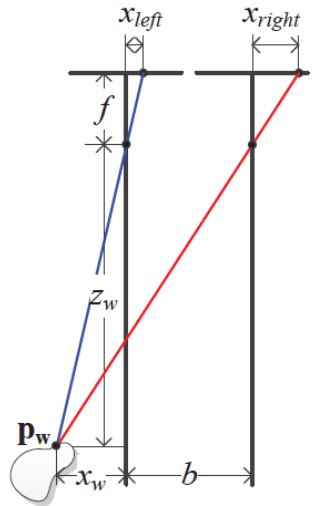


우측 영상

실체 차이 지도



(a) 투영 과정



(b) 1차원으로 단순화

왼쪽 영상에서 $\frac{x_{left}}{f} = \frac{x_w}{z_w}$

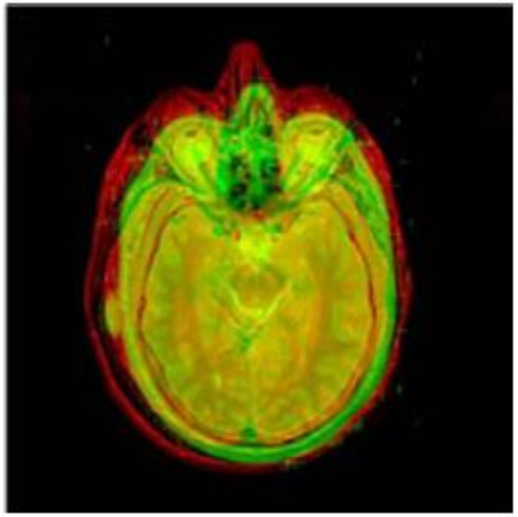
오른쪽 영상에서 $\frac{x_{right}}{f} = \frac{b + x_w}{z_w}$

$$z_w = \frac{bf}{x_{right} - x_{left}} = \frac{bf}{d}$$

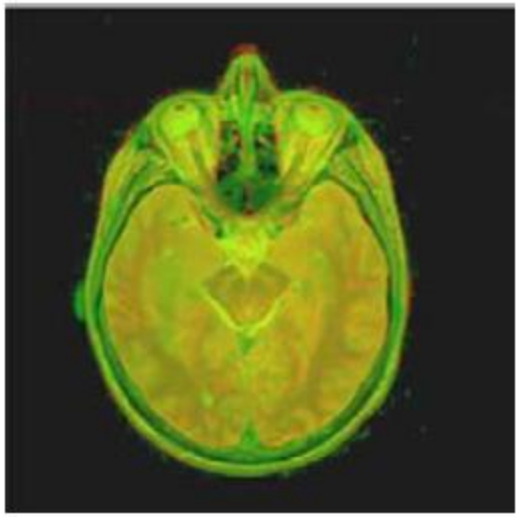
특징 추출

❖ **대응점의 활용**

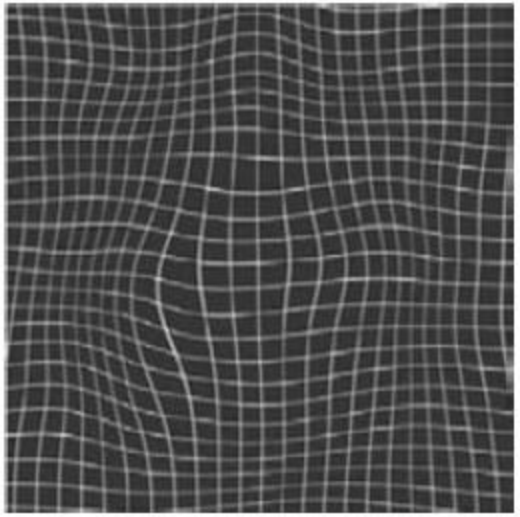
- **영상 정합**(image registration)
  - 하나의 장면을 다른 시점에서 촬영한 **두 개의 영상**이 있을 때, 하나의 영상을 다른 영상의 **좌표계**로 **변환**시켜 나타내는 것
  - 두 영상에서 대응하는 위치 쌍들을 찾아내고, 이들 위치를 대응시키는 **기하학적 변환 행렬 탐색**



두뇌 모델/
환자 MRI



정합 후 모양

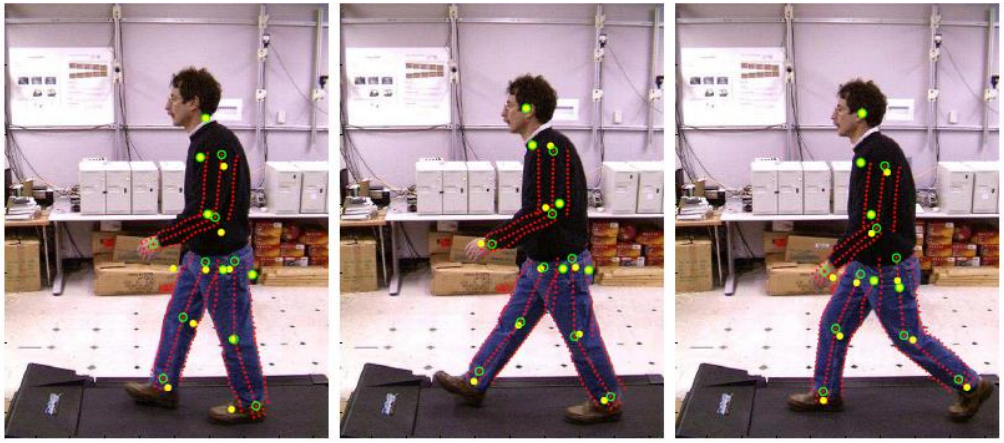


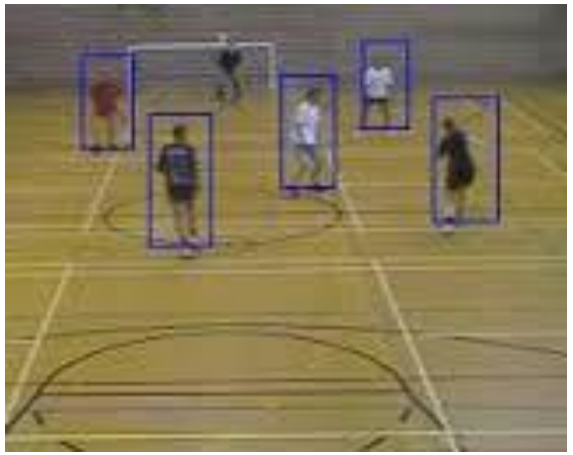
왜곡 필드

특징 추출

❖ **대응점의 활용**

- **물체 추적**(object tracking)
  - 시간적으로 인접한 두 영상에서 대응점을 찾아 추적 대상의 위치 확인





4.1 특징점

❖ **특징점**(feature point)
- 영상의 다른 곳과 **현저하게 다른 곳**

❖ **기술자**(descriptor)
- **특징점의 주변 정보**를 뽑아내어 **표현**하는 **알고리즘** 및 **표현 결과**
- 대표적인 기술자 : **SIFT**

❖ **매칭**(matching)
- 특징점의 기술자로 표현된 값들을 비교하여 **유사도**를 **계산**해 비슷한 것을 찾아내는 과정
- 비교할 대상이 많은 경우에는 효율적인 비교 알고리즘 필요

특징점

❖ 지역특징(local feature)

- 그레이 영상에서 직접 검출하는데, 다른 곳과 현저하게 차이가 나는 **특징 정보가 풍부한 곳을** 찾는 것

- **지역 특징점의 요구조건**
  - **반복성**(repeatability) : 한 영상에서 특징점으로 검출된 것은 다른 영상들에서도 유사한 특성을 갖는 특징점으로 검출
  - **분별력**(distinctiveness) : 물체의 다른 곳과 충분히 구별
  - **지역성**(locality) : 특징점 주변의 작은 영역에서 특징 정보가 충분
  - **정확성**(accuracy) : 특징점의 위치를 정확히 결정 가능
  - **효율성** : 특징점이 너무 많이 검출되지 않으면서 짧은 계산 시간

4.2 영상 피라미드와 스케일 공간

❖ **영상 피라미드**(image pyramid)

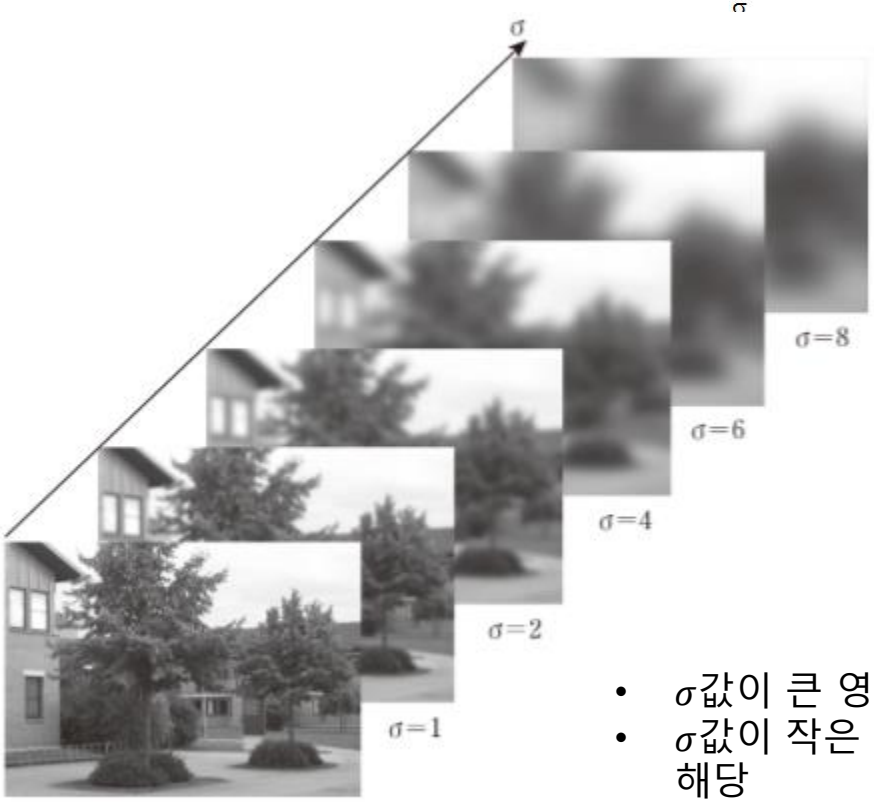
- 영상의 가로, 세로 **길이**를 각각 **1/2**로 줄여가면서 생성한 일련의 이미지
- **크기 변화**에 **대응**할 수 있는 **특징점 검출 가능**
- 영상의 크기가 1/4로만 줄어드는 제약

그림 9.26 영상 피라미드

영상 피라미드와 스케일 공간

❖ **스케일 공간**(scale space)

- 멀리 떨어져 있는 물체가 희미하게 보인다는 점에 착안
- 영상의 크기를 줄이는 것이 아니라 **가우시안 필터의 표준편차** σ값을 **점점 키워가면서** 여러 개의 **영상을 만드는 것**



- σ값이 큰 영상에서 검출되는 특징은 크기가 큰 특징에 해당
- σ값이 작은 영상에서 검출되는 특징은 크기가 작은 특징에 해당

그림 9.27 스케일 공간 σ는 적용한 가우시안 필터의 표준편차를 나타낸다.

4.3 블롭 검출

❖ 블롭 검출(blob detection)

- 라플라시안 필터는 블롭(위치, 크기)을 검출하는 역할

$$LOG(y,x,\sigma) = \nabla^2 G(y,x,\sigma) = \frac{1}{\pi\sigma^2}\left(\frac{x^2+y^2-2\sigma^2}{2\sigma^2}\right)e^{\frac{-x^2+y^2}{2\sigma^2}}$$

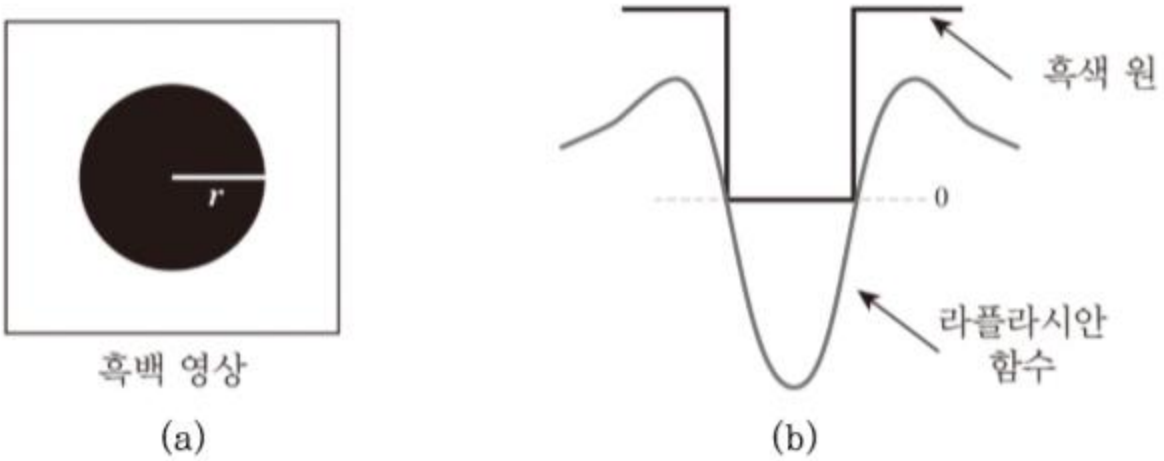


그림 9.28 라플라시안 함수의 효과

(a) 원의 중심에 라플라시안 함수의 필터를 씌워 컨볼루션을 할 때, 가장 큰 값을 얻기 위해서는 (b)와 같이 라플라시안 함수의 값이 0이 되는 부분이 흑색원의 경계가 된다.

블롭 검출

❖ 블롭 검출(blob detection)

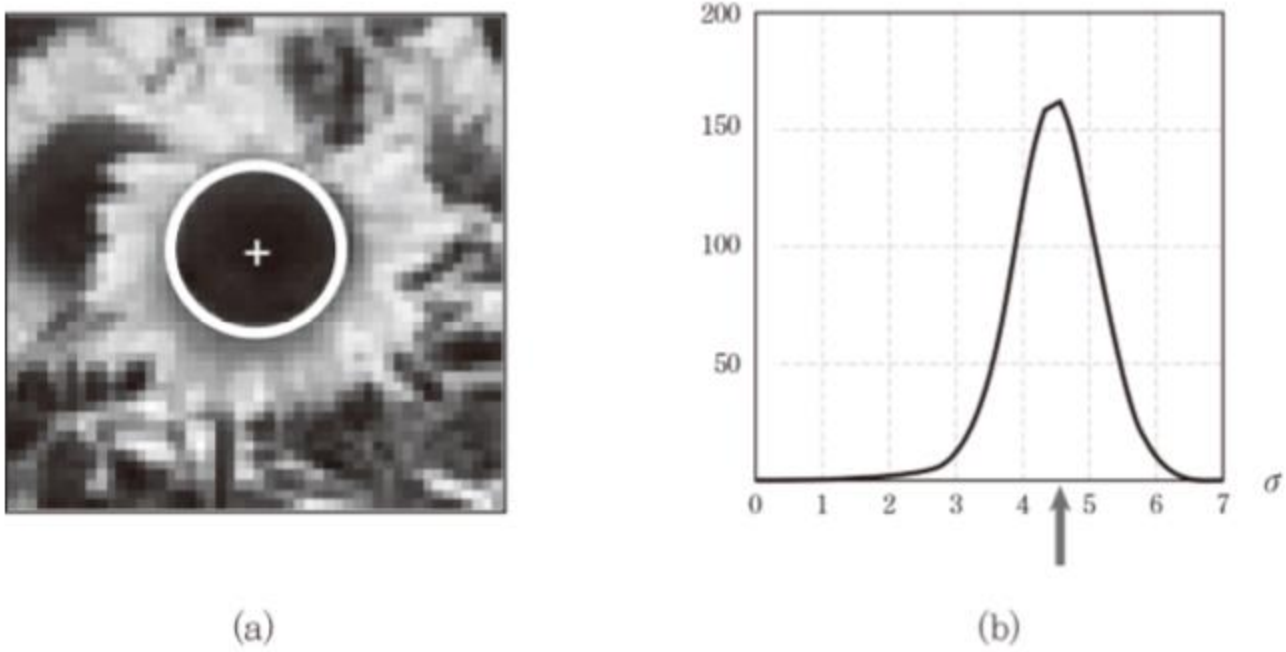


그림 9.29 σ값에 따른 LOG 컨볼루션 값의 변화

(b)는 (a)의 '+' 표시된 위치에서, σ를 변경하면서 LOG 컨볼루션을 계산한 결과 값을 나타낸다.

4.4 SIFT 특징점 검출

❖ **SIFT**(Scale-Invariant Feature Transform)

- 1999년 로우(David G. Lowe)가 개발한 대표적인 지역특징 추출 방법
- **키포인트**(keypoint)
  - SIFT에서 특징점
- 스케일 공간과 피라미드 구조 사용
  - **옥타브**(octave)
    - 같은 크기의 스케일 공간
    - 보통 5~6개의 영상 포함
  - 인접 영상과의 σ값의 차이 k 비율
    - $2^{1/3} \approx 1.26$
    - $\sigma = 1.6$
  - 다음 옥타브의 첫번째 영상
    - $k^2\sigma$의 영상에서 다운샘플링 (downsampling)
      - » 하나 건너 하나 선택
  - 영상 크기가 4×4가 될때 까지 반복

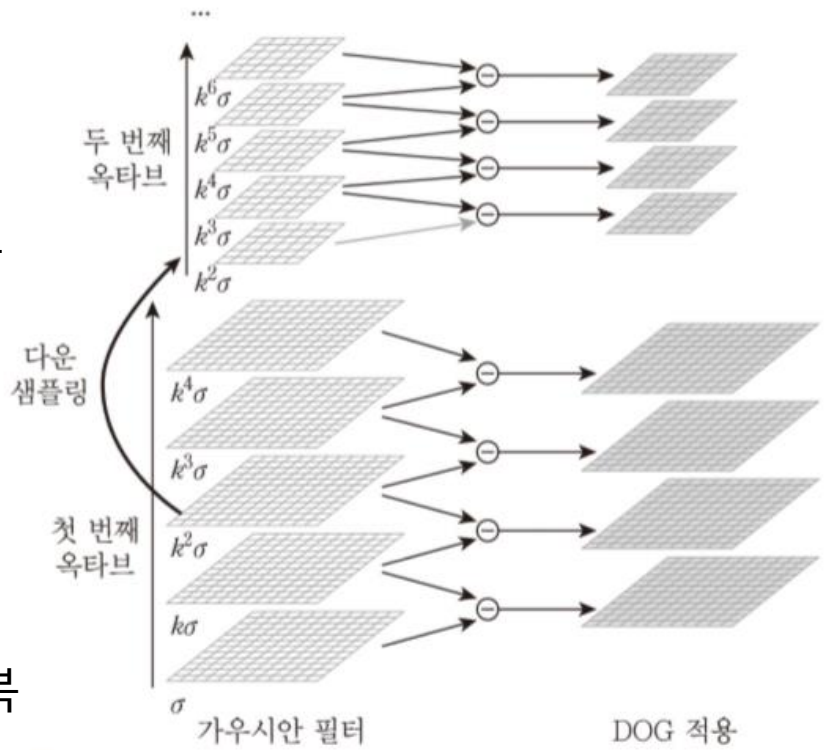


그림 9.30 SIFT의 스케일 공간과 피라미드

SIFT 특징점 검출

❖ SIFT

- 옥타브 내의 인접 영상에 대해서 DOG 계산
  - LOG를 하는 효과
  - 피라미드 구축과정에서 가우시안 필터 적용
- **키포인트 선택**
  - 위와 아래 DOG 영상들을 포함해서 이웃한 26개의 값과 비교
  - '×' 위치의 값이 최소나 최대가 되면 극점으로 선택

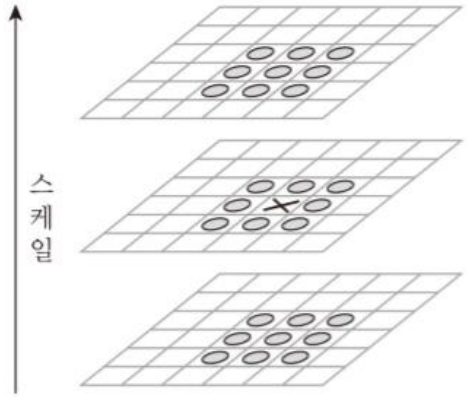
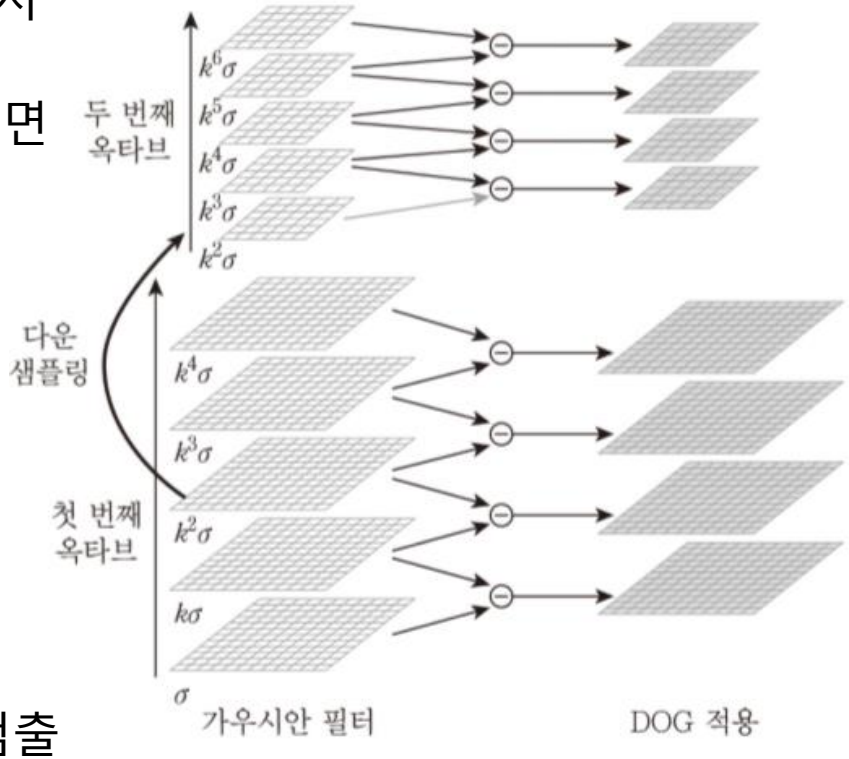


그림 9.31 키포인트 검출



- 다양한 크기의 블롭을 찾는 특징점 검출

SIFT 특징점 검출

❖ **SIFT 키포인트(특징점) 검출결과**

- 특징점 위치 : 원의 중심
- 특징점 크기 : 원의 크기

SIFT 특징점 검출

❖ **특징점 추출 방법**

- SIFT
- 해리스-라플라스(Harris-Laplace)
- SURF(Speeded-Up Robust Features)
- FAST
- ORB
- BRISK
- …

4.5 특징 기술자

❖ 특징 기술자

- 관심점(interest point, 특징점)의 특징을 추출한 정보를 기술한 것 또는 이러한 정보를 추출하는 알고리즘

- 기술자의 요구 조건
  - 관심점들을 잘 구별할 있는 **분별력**(discriminating power)
  - 회전, 크기변화, 이동과 같은 기하학적 변화에 대해서 영향을 받지 않는 **불변**(invariant)
  - 조명변화, 잡음, 가림에 대해서도 **강건**(robust)

- 기술자의 종류
  - **SIFT의 기술자**
  - SIFT의 변형인 **PCA-SIFT**와 **GLOH**
  - **모양 콘텍스트(shape context)**
  - **BRIEF**
  - **ORB**
  - **BRISK** 등

SIFT 특징 기술자

❖ SIFT 특징 기술자

- 회전변환에 불변인 기술자 생성
- 키포인트(관심점)의 위치와 크기가 주어지면 **지배적 방향**(dominant direction) 결정
- 키포인트를 중심에 두고 키포인트의 크기에 비례하는 크기의 가우시안 윈도우 적용
- 윈도우와 중첩되는 위치의 각 화소에 대해 **그레디언트**(gradient)를 계산 해당 위치의 가우시안 필터값을 곱함
- 그레디어트를 10도 구간으로 분할, 히스토그램 작성
- 지배적 방향으로 회전

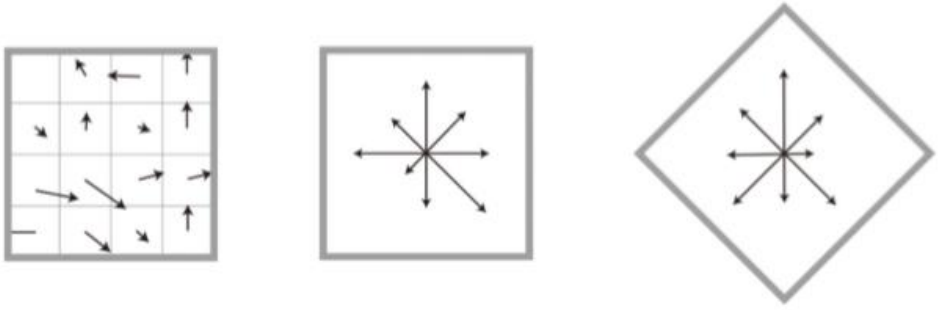


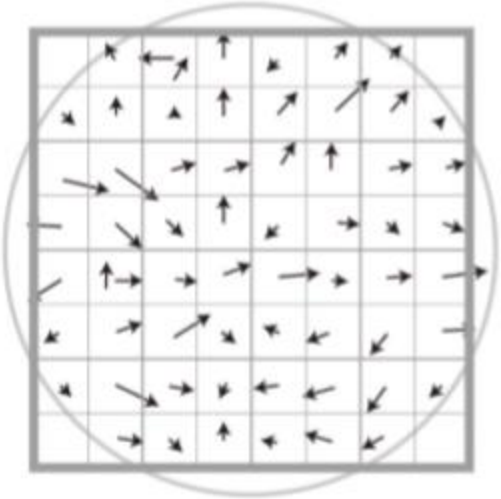
그림 9.32 SIFT에서 지배적 방향 선정

(a) 그레디언트 벡터 (b) 그레디언트 히스토그램 (c) 지배적 방향이 위쪽을 향하도록 회전한 결과

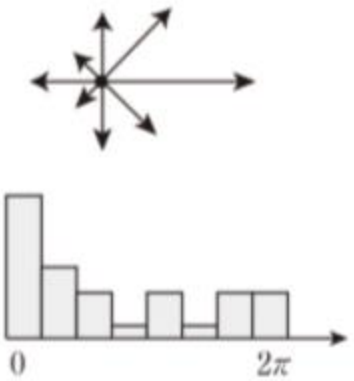
SIFT 특징 기술자

❖ **SIFT 특징 기술자**

- **SIFT 기술자의 특징 기술**
- 전체 윈도우를 4×4블록으로 나누고, 각 블록에 대해서 8단계로 양자화하여 히스토그램 계산
- 각 블록의 히스토그램을 모아둔 형태로 $4 \times 4 \times 8 = 128$ 차원의 벡터
- 크기, 방향, 광도 변화에 불변성 특성 추출



(a)



각도 히스토그램

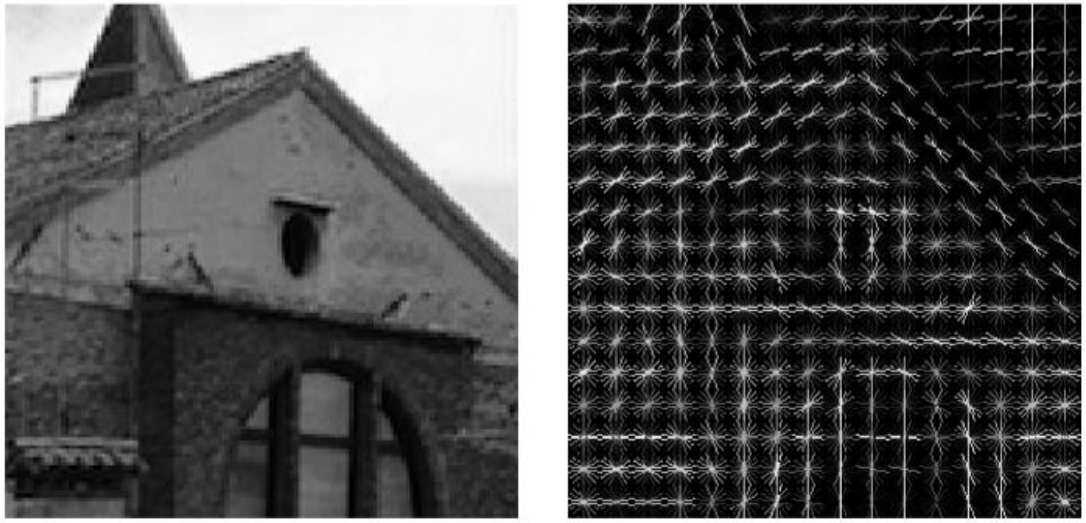
(b)

그림 9.33 SIFT 기술자의 특징 기술

4.6 HOG 기술자

❖ HOG 기술자

- HOG(Histogram of Oriented Gradients, 방향성 그레이언트의 히스토그램)
- **사람의 형태**나 **보행자**를 **검출**하는 데 주로 사용되는 영상의 기술자
- 영상을 일정 크기의 **블록**으로 나누어 **그레디언트**를 **계산**한 다음, 그레디언트를 이용하여 해당 블록의 **지역적 히스토그램**을 생성하고, 지역적 히스토그램을 **이어붙여 1차원 벡터**로 된 기술자 생성



(a) (b)

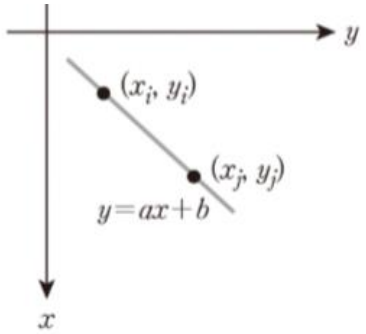
그림 9.34 HOG 기술자

(a) 원 영상 (b) 셀별 그레디언트 히스토그램

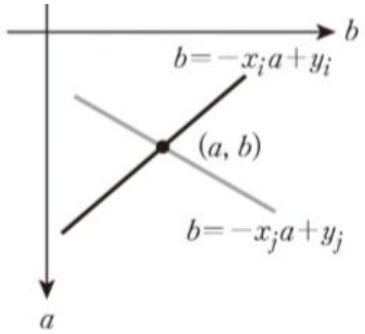
4.7 허프 변환

❖ **허프 변환** (Hough transform)

- 영상에서 **직선**이나 **곡선**을 찾는 데 사용되는 기법
- 칼러 영상 이라면 **그레이 영상**으로 변환한 다음,
  Canny 연산자 등을 통해서 **윤곽선**에 해당하는 **위치**들이 **점**으로 표시된 **이진 영상**으로 변환한다고 전제



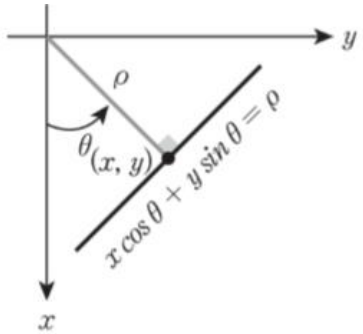
(a)



(b)

그림 9.35 직선의 표현

(a) xy —평면 (b) ab —평면



(a)

(b)

그림 9.36 $\rho\theta$ — 공간에서 직선의 표현

(a) 점 (x,y) 와 (ρ,θ) 의 관계 (b) $\rho\theta$ — 공간

허프 변환

❖ **허프 변환** – cont.

- 영상에서 **직선**을 찾는 방법
  - 특징공간을 일정간격의 격자로 나누고
    해당 격자에 지나가는 것들은 동일한 교차점을 갖는다고 간주
  - 많은 곡선이 지나간 격자들 만을 선택해서 해당 격자에 해당하는 직선을 추출

- 방정식으로 표현되는 다른 곡선들을 찾는 데도 사용 가능

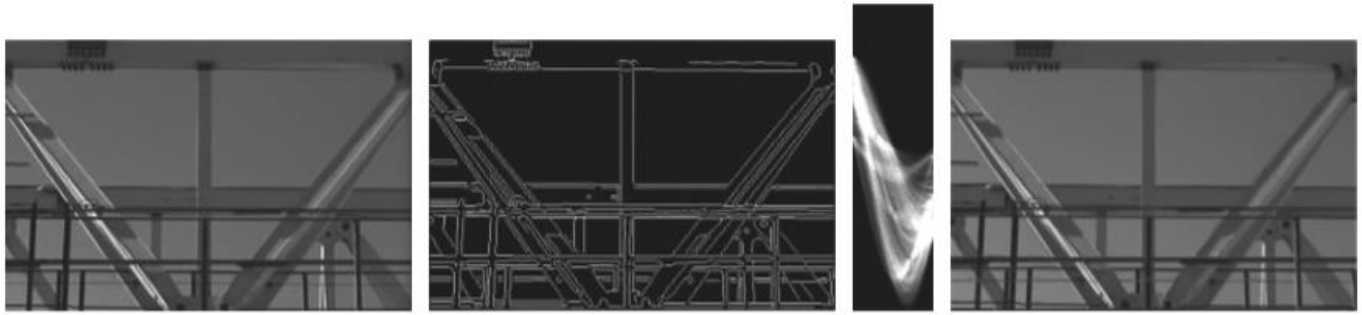


그림 9.37 허프변환 적용 예

(a) 원본 영상 (b) Canny 연사자를 적용한 윤곽선 이진 영상 (c) $\rho\theta$ —공간 표현 (d) 식별된 직선

4.8 매칭

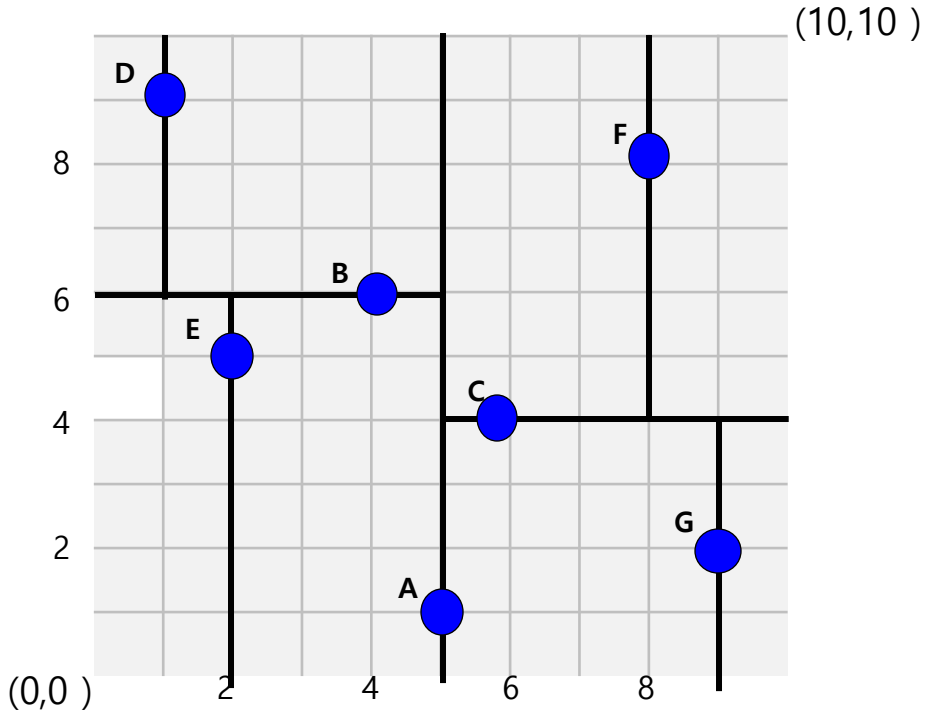
❖ **매칭**(matching)

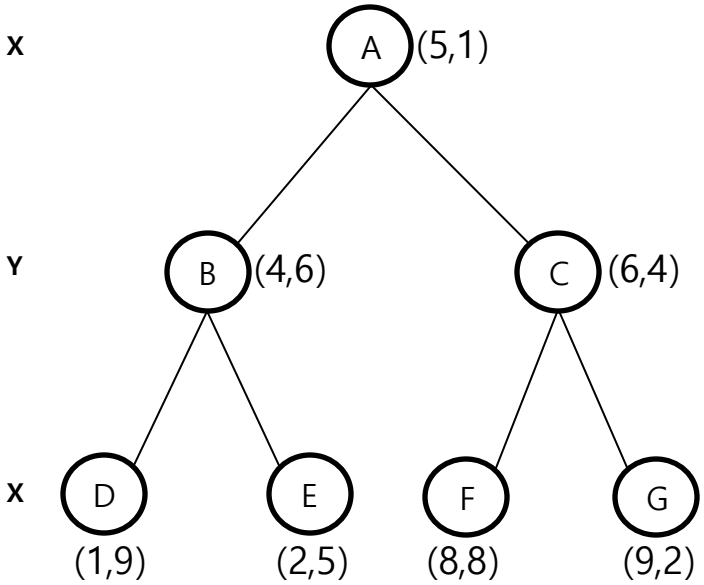
- 대상들을 비교하여 유사한 것들을 찾아내는 것
- 특징점이 추출되고 이에 대한 기술자가 벡터로 주어지면, 특징점들은 고차원 공간에 있는 점에 대응

- **인덱싱 구조 사용**
  - **k-d 트리**(k-d tree)
    - 공간을 각 차원에 직교하는 공간으로 분할하여 공간을 쉽게 찾도록 도와주는 자료구조
    - 비교적 낮은 차원(10차원 이내)에는 효과적
  - **지역민감 해싱**(locality sensitive hashing) 방법
    - 고차원 데이터에 대해 빠른 근사적 검색이 가능한 기법
    - 유사한 데이터가 같은 버킷(bucket)으로 대응될 확률이 큰 해시 함수(hash function)를 사용

매칭

- **k-d 트리**(k-d tree)
  - [Bently, 1975]
  - 고차원 데이터인 경우 전체 데이터에 대한 비교를 할 만큼 느림

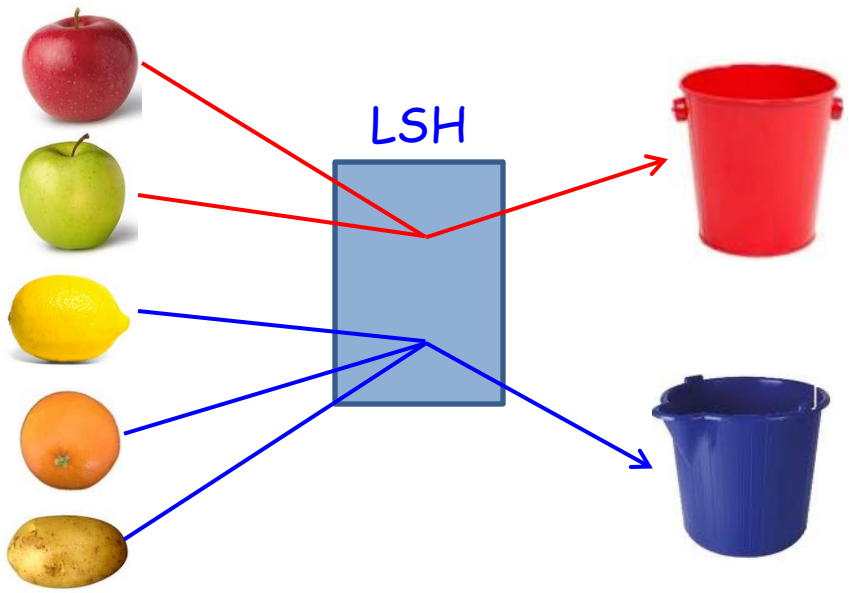




매칭

❖ **지역민감 해싱**(locality sensitive hashing)

- 유사한 데이터는 같은 버킷(bucket)으로 높은 확률로 해싱
- 차이가 큰 데이터는 같은 버킷으로 낮은 확률로 해싱

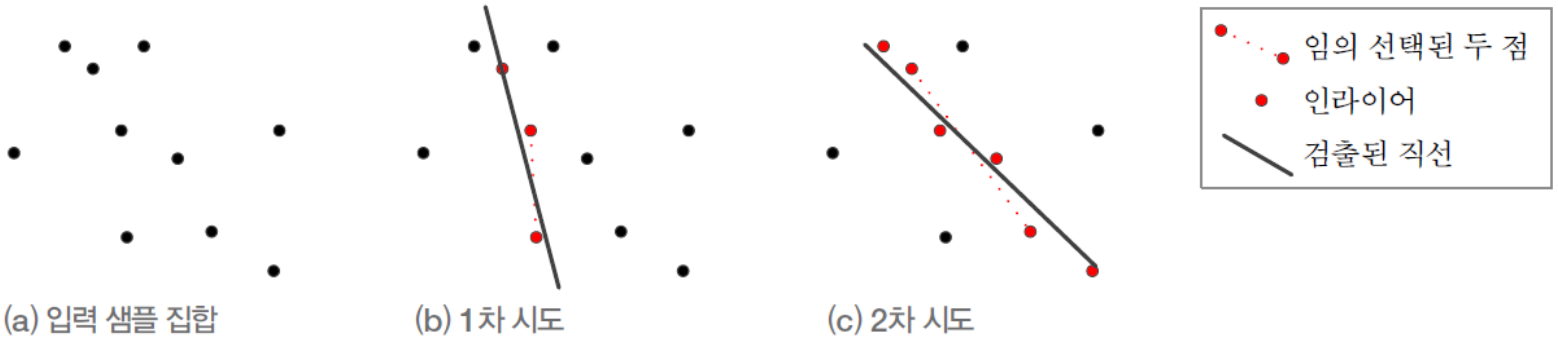


매칭

❖ 기하정렬과 RANSAC

- 동일 장면을 다른 시점에서 획득된 두 영상에서 대응되는 객체의 기하학적 변환 관계
- 한 영상을 기하하적 변환 행렬에 곱하면 다른 영상이 근사하게 만드는 변환 찾기

- **RANSAC**(Random Sample Consensus) 알고리즘
  - 주어진 여러 데이터들 중에서 임의로 몇 개를 선택하여 함수를 근사한 다음, 해당 함수와 부합하는 데이터들을 확인
  - 해당 함수가 얼마나 바람직한지 평가하거나, 부합하는 데이터들을 포함시켜 함수를 개선



매칭

❖ 기하정렬과 RANSAC

- **대응점 문제**에 대한 RANSAC 적용
  - 두 영상에 대한 대응점의 쌍들의 데이터를 입력으로 사용
  - **대응점 쌍**들 중에서 무작위로 **세 쌍**을 선택
  - 세 대응점에서 한 영상의 좌표값은 입력으로 다른 영상의 좌표값은 출력으로 만들어주는 **변환 행렬 T 계산**
    - 최소제곱법(least mean square method), 최소제곱중간값법(least median square method) 등 사용
  - **나머지 대응점 쌍**의 변환행렬 T의 변환관계를 **만족하는지 평가**
  - 다시 무작위로 세 쌍의 대응점들을 선택한 다음에 위의 과정 **반복**
  - 여러 번 반복하여 **가장 좋은 것**을 **변환행렬 사용**
  - 영상들에 대해서 대응점의 쌍들과 변환 행렬을 찾으면, 영상들의 이어붙이기(stitching) 가능

매칭

- 영상 이어붙이기
  - 대응점 찾기
  - 변환행렬 찾기

5. 컴퓨터 비전 대상 문제

❖ 동영상 처리 기술

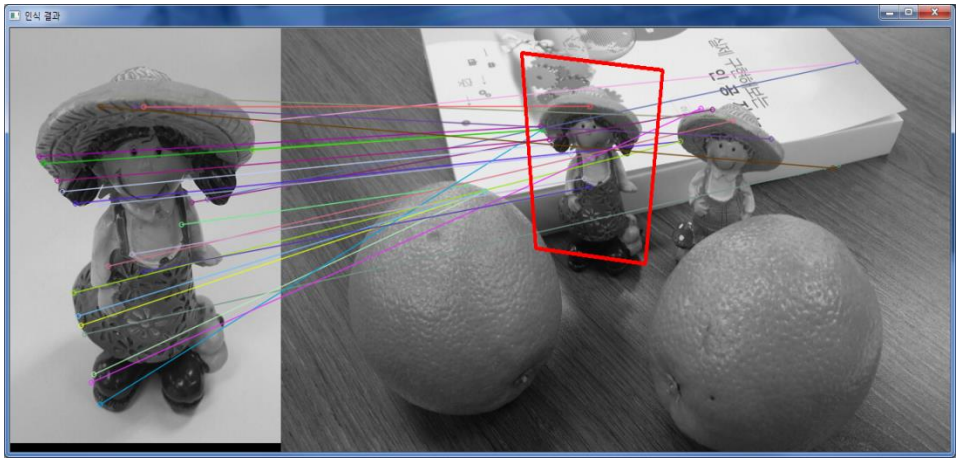
- 정지영상이 연속해서 있는 것으로 보고 처리
- 물체의 움직임 추적
  - 인접한 두 장의 영상에서 움직임을 검출하는 **광류(optical flow)** 관련 기술
  - 예측 모델을 사용하여 움직임을 추적하는 기술

- 광류(optical flow)
  - 영상의 물체들의 속도 분포

컴퓨터 비전 대상 문제

❖ 인식문제

- **사례 인식**(instance recognition)
  - 특정 물체가 영상에 있는지 찾는 것
  - SIFT와 같은 우수한 지역특징이 개발되면서 높은 성능 구현



- **범주 인식**(category recognition)
  - 영상 속에 나타나는 물체가 어떤 범주에 속하는지 결정하는 문제
  - 범주 내의 변화가 매우 크기 때문에 아직 만족스러운 결과를 얻지 못함

6. 객체 위치 검출 및 개체 인식

❖ **객체 위치 검출 및 개체 인식을 위한 딥러닝 모델**

- R-CNN 모델
  - R-CNN, Fast R-CNN, Faster R-CNN
- YOLO 모델
- SSD 모델

6.1 R-CNN 모델

❖ **R-CNN 모델**

- 먼저 객체를 포함하고 있을 것 같은 영역을 찾기 위해 기존의 **영역 제안알고리즘** 적용
- 추천된 각 영역의 특징 추출을 위해, AlexNet의 변형된 형태인 CNN 모델 사용

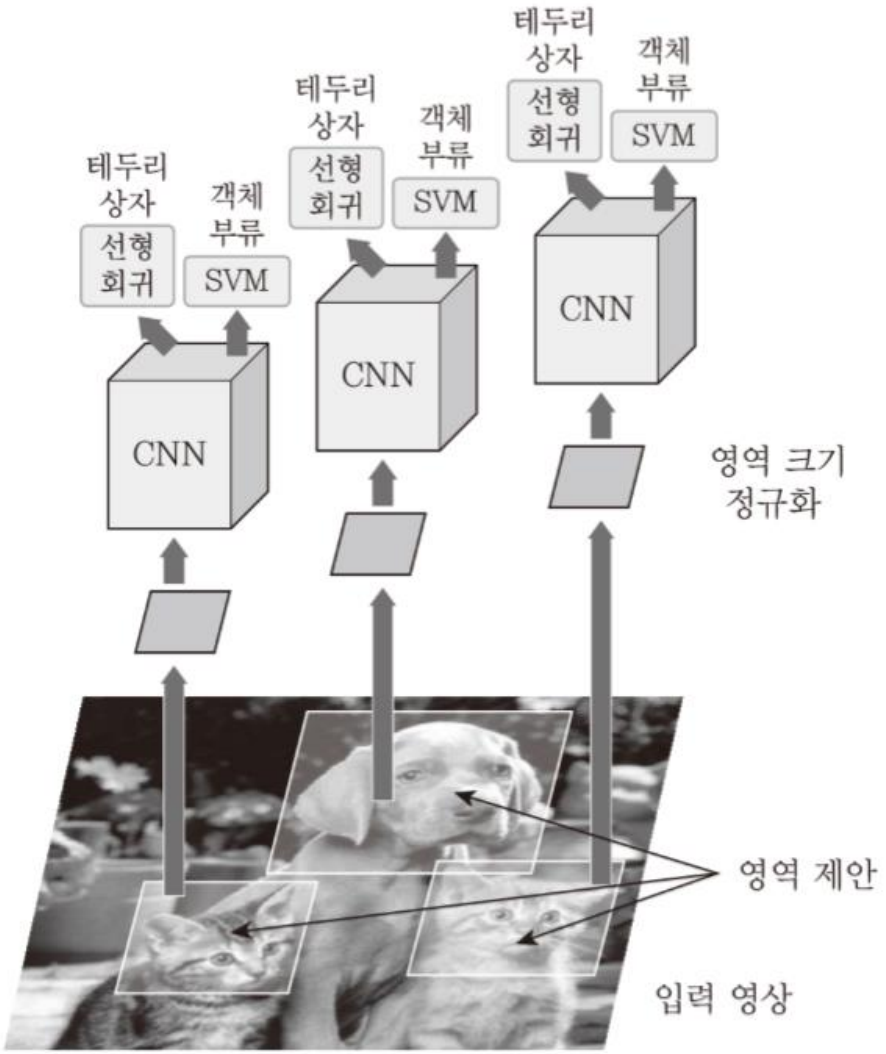


그림 9.39 R-CNN 모델

R-CNN 모델

❖ **Fast R-CNN 모델**

- **특징 지도**에서 **영역 제안 알고리즘**이 추천한 물체 영역들에 대한 대응 위치를 찾아, 관심 영역으로 선택
- **완전 연결층의 출력**은 관심 영역 내의 객체를 분류하는 **소프트맥스 분류기**와 추천된 객체 영역을 미세 조정하는 **선형 회귀 모델**로 각각 전달

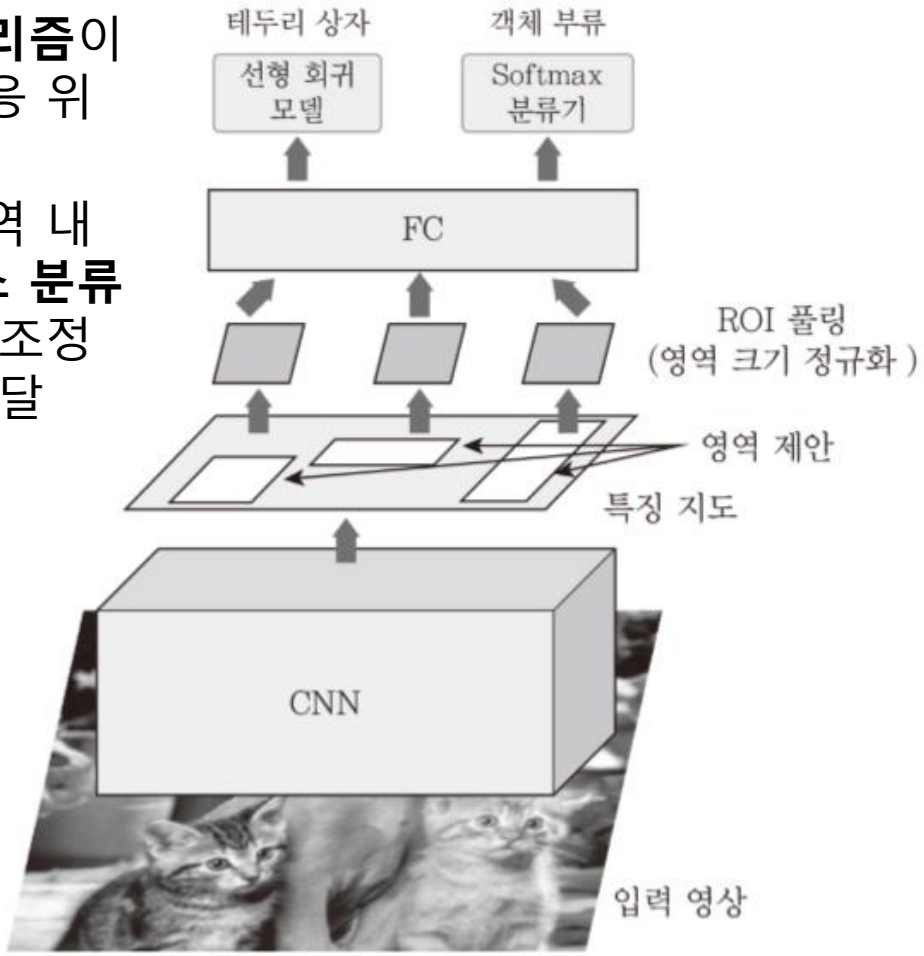


그림 9.40 Fast R-CNN 모델

R-CNN 모델

❖ **Faster R-CNN 모델**

- 특징 지도로부터 직접 객체 영역을 찾는 **영역 제안 망**을 내부에 포함

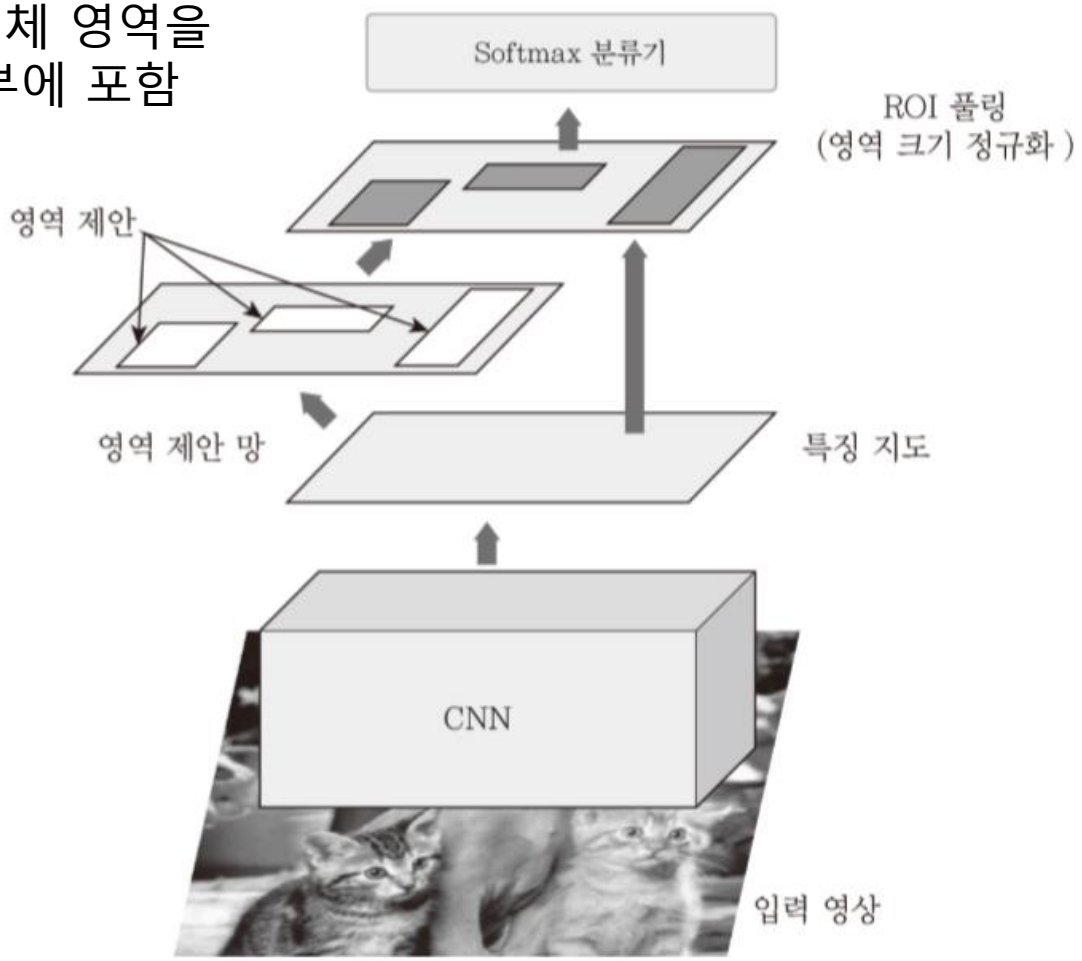


그림 9.41 Faster R-CNN 모델

R-CNN 모델

❖ **Faster R-CNN의 영역 제안 망**

- 특징 지도에 대해 여러 크기의 앵커에 대해 객체의 유무 평가

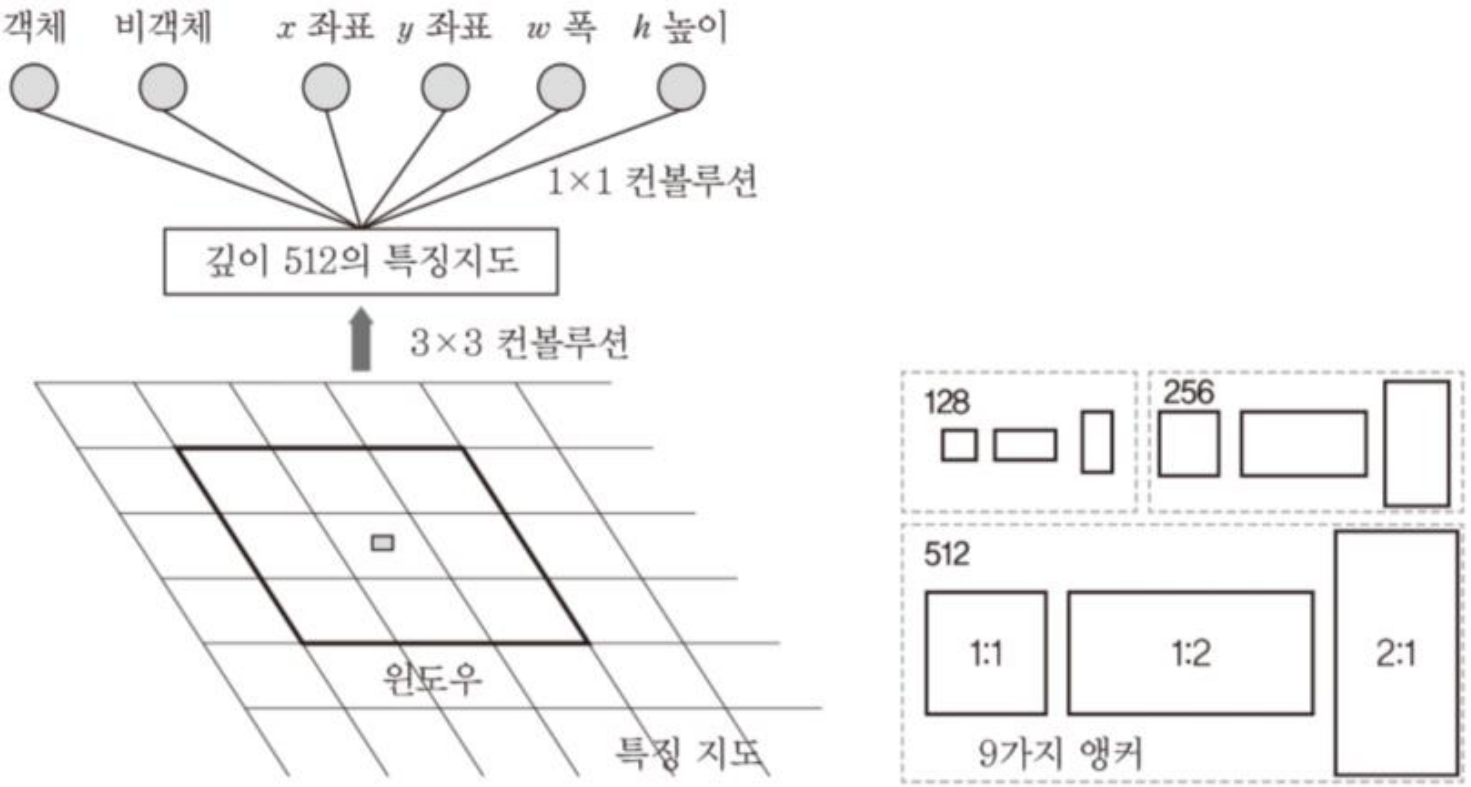


그림 9.42 Faster R-CNN의 영역 제안 망

6.2 YOLO 모델

❖ YOLO 모델

- 실시간으로 객체를 감지하고 인식하는 모델

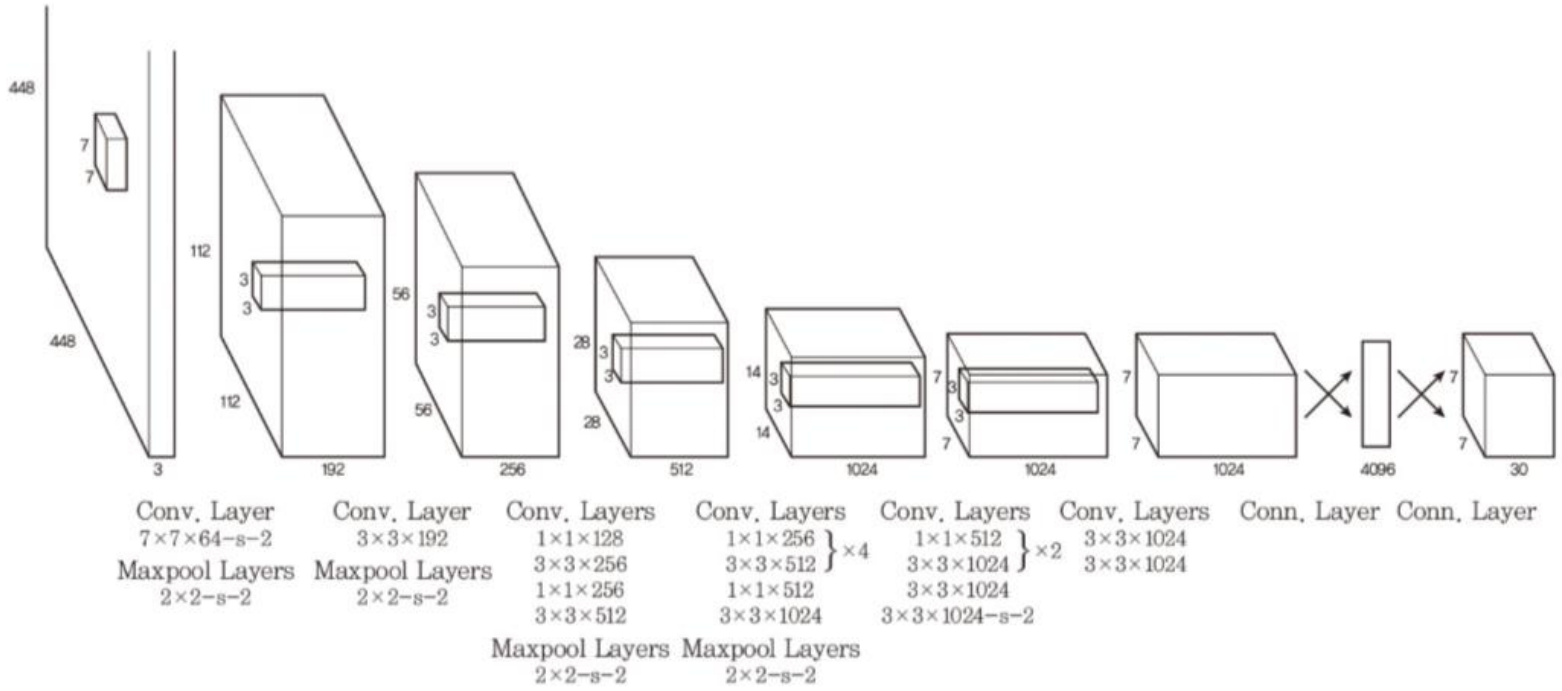


그림 9.43 YOLO의 구조[출처: Redmon 등 2015]

- 개선된 모델
  - YOLO Fast, YOLOv2, YOLOv3

6.3 SSD 모델

❖ SSD 모델

- 각 영상에 해서 **고정된 크기**의 **테두리 상자**들을 지정하여 객체에 대응하는 테두리 상자와 해당 상자에서 객체의 부류를 잘 찾을 수 있도록 학습
- 실시간 객체 위치 식별 및 인식

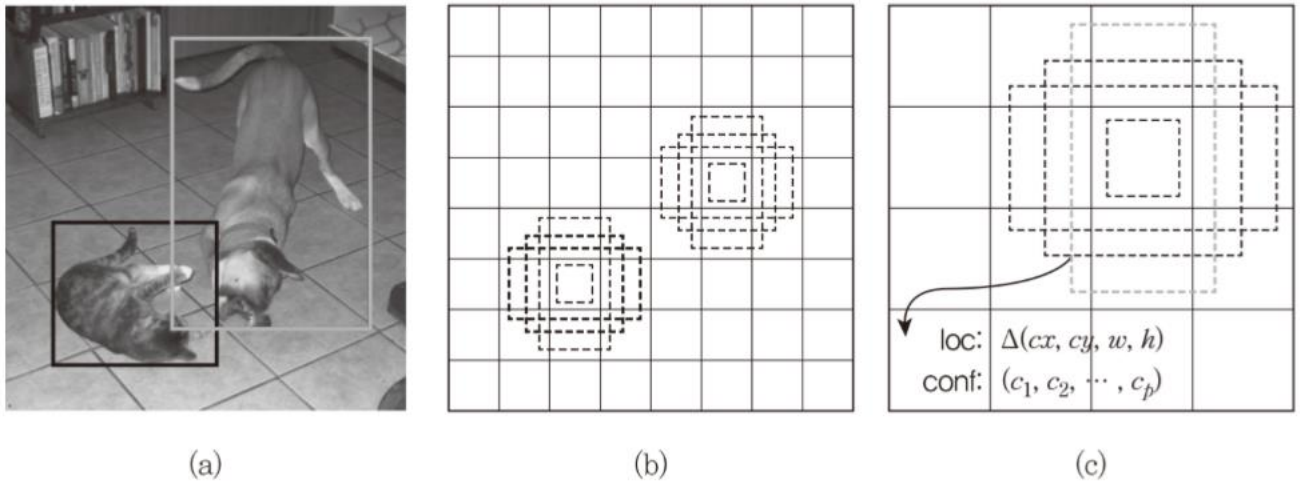


그림 9.44 기준 테두리 상자와 기본 테두리 상자[출처: Liu 등, 2015]

(a) 객체(고양이, 개)의 위치를 표현한 기준(ground truth) 테두리 상자 (b) 8×8 특징 지도와 각 위치별 4개의 기본(default) 테두리 상자 (c) 4×4 특징지도와 '강아지'에 매칭된 기본 테두리 상자에 대한 위치 정보 $\Delta(cx, cy, w, h)$ 와 각 부류별 신뢰도 $(c_1, c_2, \cdots, c_p)$

SSD 모델

❖ SSD 모델의 구조

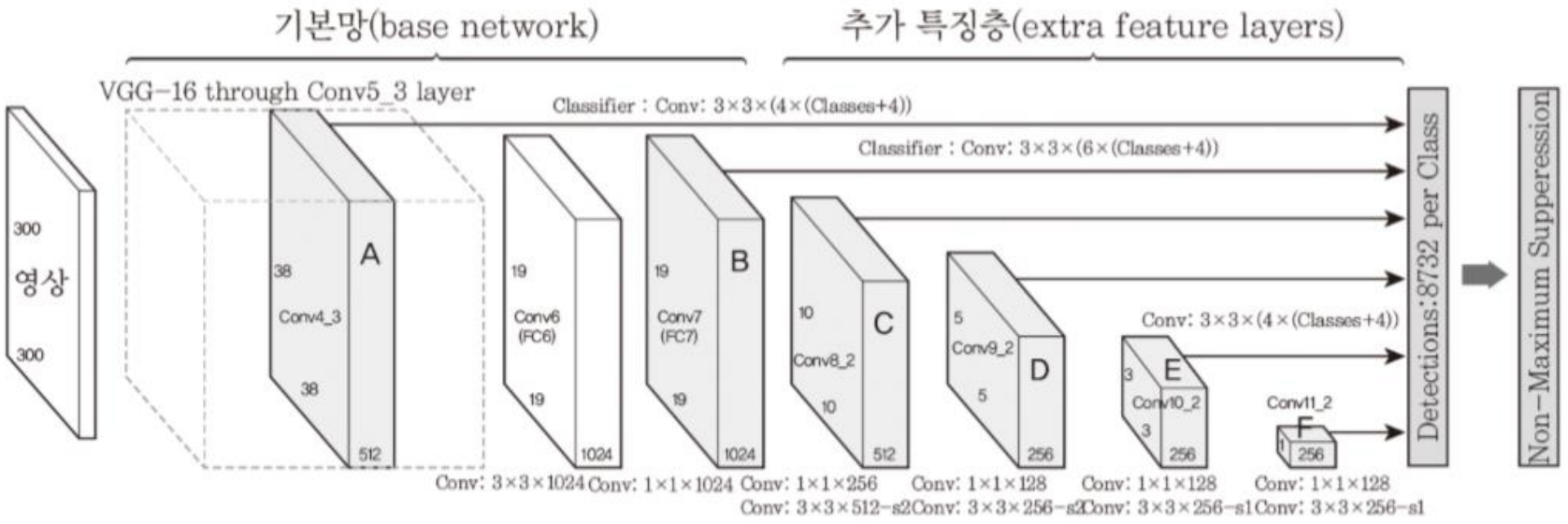


그림 9.45 SSD의 구조[출처: Liu 등, 2015]

7. 의미적 영역 분할

❖ **의미적 영역 분할** (semantic segmentation)
- 각 화소가 어떤 부류에 속하는지 결정하는 것
- 영역 분할을 하면서 해당 영역에 있는 객체의 부류도 함께 결정하는 것

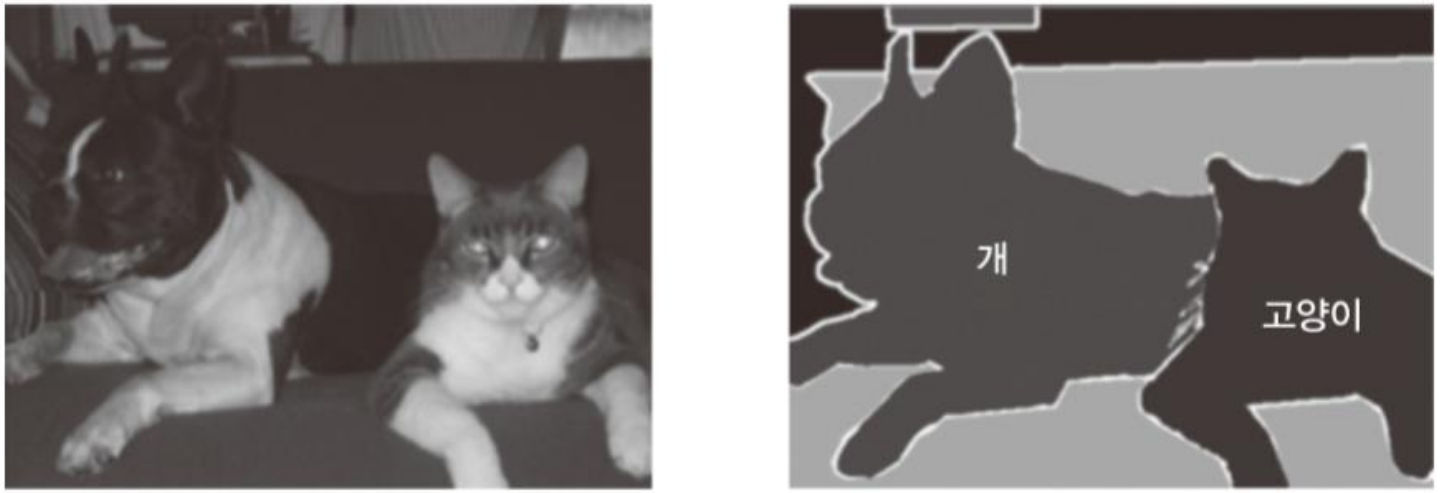


그림 9.46 **의미적 영역 분할** (a) 원본 영상 (b) 객체 영역과 객체 부류

의미적 영역 분할

❖ **FCN**(fully convolutional network) **모델**
- 의미적 영역 분할을 위한 딥러닝 모델

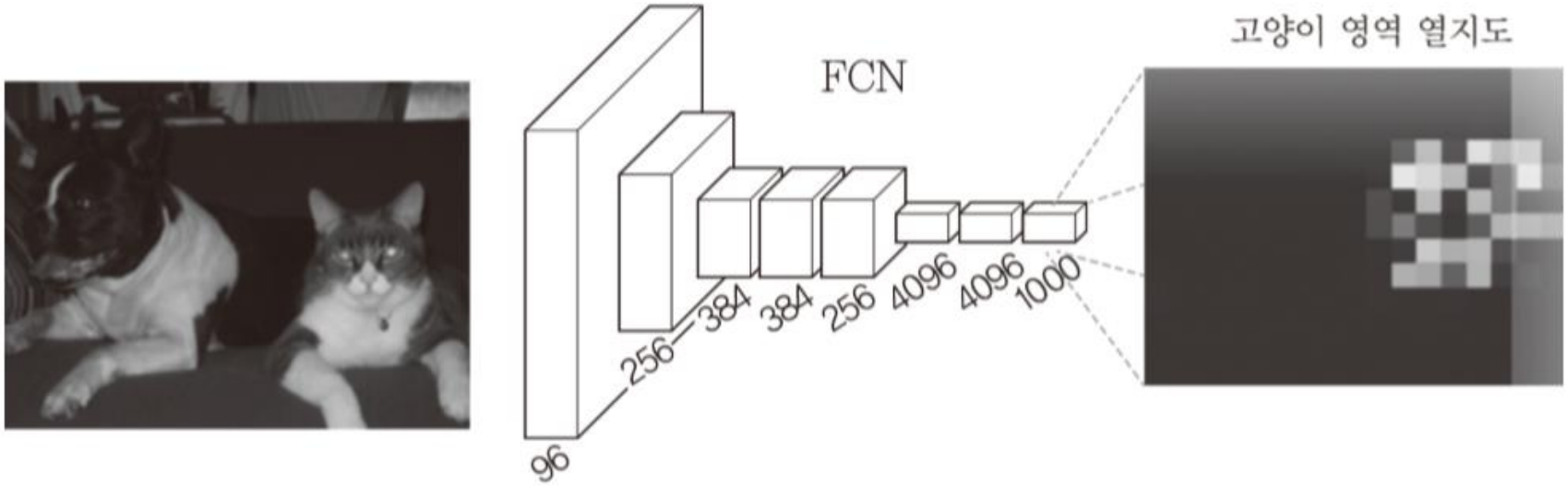


그림 9.47 FCN의 구성[출처: Long 등 2015]

8. 딥러닝 응용

❖ 영상 주석달기

- 영상이 주어지면 영상의 내용을 묘사하는 문장을 만들어 내는 것
- **입력 영상**에 대해 **CNN**을 적용하여 **맥락정보**를 **추출**하고, 이를 **초기 정보**로 사용하여 **LSTM 재귀 신경망**이 문장 생성

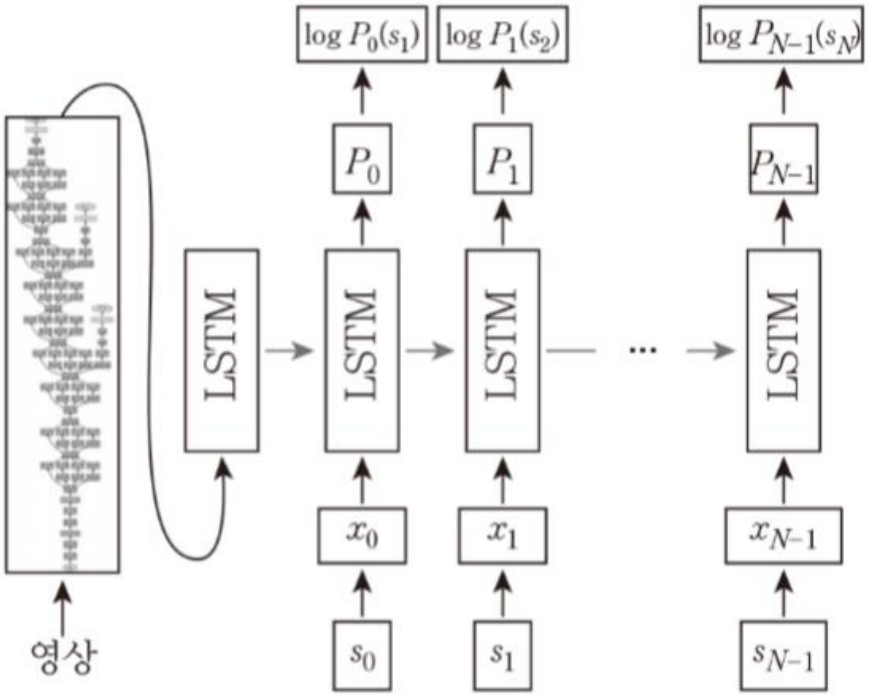


그림 9.50 영상 주석달기 신경망의 구조[출처: Karpathy 등, 2015]

8.1 영상 주석달기

❖ 영상 주석달기

A man skiing down a snow covered slope.

A group of giraffe standing next to each other.

그림 9.51 딥러닝에 의한 영상 주석달기의 예

8.2 예술작품 화풍 그림 생성

❖ **예술작품 화풍 그림 생성**

- CNN 모델에 영상을 입력으로 넣어주면, CNN 망의 각 층에서 여러 채널의 특징지도가 생성
- 같은 층에 있는 채널 간의 상관계수가 화풍과 관계가 있다는 성질을 이용
- 화풍의 특징에 대한 유사도와 내용에 대한 유사도를 반영한 손실함수 정의
  - 모델의 가중치를 수정하는 것이라 입력 영상을 수정

예술작품 화풍 그림 생성

La muse

변환

그림 9.52 화풍에 따른 사진의 그림 변환

왼편 사진을 피카소 작품 'La Muse' 화풍의 그림으로 변환한 것.