스마트홈 IoT가전기기 제어를 위한 객체인식 모델구현

사전

딥러닝을 위한 파이썬 프로그래밍과 영상처리 개념 이해(12)

KETI 한국전자기술연구원
Korea Electronics Technology Institute

학습내용

- 영상분할 및 특징추출
- K-최근접 이웃 분류기
- 허프 변환(Hough Transform)
- 영상 처리 응용 사례

학습목표

- 허프 변환 방법을 이용하여 영상의 특징을 추출할 수 있다.

영상분할 및 특징추출

영상분할 및 특징추출

◆ 영상분할 및 특징추출

- 영상분할
  - → 입력 영상에서 픽셀 단위로 배경 및 객체를 구하는 작업
  - → 영상의 밝기 값, 컬러, 텍스처, Gradient 등 정보를 이용
- 특징추출
  - → 추적이나 식별의 목적으로 다양한 영상 처리 알고리즘에서 활용
  - → 코너점, 경계선 정보 등

K-최근접 이웃 분류기

K-최근접 이웃 분류기

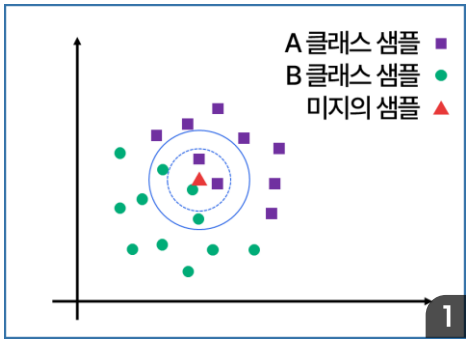
◆ K-최근접 이웃 분류기의 이해

- 최근접 이웃 알고리즘
  - → 기존에 가지고 있는 데이터들을 일정한 규칙에 의해 분류된 상태에서 새로운 입력 데이터의 종류를 예측하는 분류 알고리즘
  - → 학습 클래스의 샘플들과 새 샘플의 거리가 가장 가까운 (Nearest)클래스로 분류

가장 가까운 거리

  - 미지의 샘플과 학습 클래스 샘플 간의 유사도가 가장 높은 것을 의미
  - 유클리드 거리(Euclidean distance), 해밍 거리(Hamming distance), 차분 절댓값

- K-최근접 이웃 분류(K-Nearest Neighbors: k-NN)
  - → 학습된 클래스들에서 여러 개(K)의 가까운 이웃을 선출하고 이를 이용하여 미지의 샘플들을 분류하는 방법
  - → k가 3일 경우
    - 미지 샘플 주변 가장 가까운 이웃 3개 선출
    - 이 중 많은 수의 샘플을 가진 클래스로 미지의 샘플 분류
    - A 클래스 샘플 2개, B 클래스 샘플 1개 → A 클래스 분류
  - → k가 5일 경우
    - 실선 큰 원내에 있는 가장 가까운 이웃 5개 선출
    - 2개 A 클래스, 3개 B 클래스 → B 클래스로 분류



보라색 네모 샘플 > **빨간색 세모 샘플**

K-최근접 이웃 분류기

k-NN을 위한 KNearest 클래스의 이해

예제 10.3.1

```
01  Import numpy as np, cv2
02
03  def draw_points(image, group, color):
04          for p in group
05                  pt = tuple(p.astype(int))
06                  cv2.circle(image, pt, 3, color, cv2.FILLED)
07
08  nsample = 50
09  traindata + np.zeros((nsample*2, 2), np.float32)
10  label = np.zeros((nsample*2, 1), np.float32)
11
12  cv2.randn(traindata[nsample:], 150, 30)
13  cv2.randn(traindata[nsample:], 250, 60)
14  Label[:nsample], label[nsample:] = 0, 1
15
16  K = 7
17  Nn = cv2.ml.KNearest_create()
18  Knn.train(traindata, cv2.ml.ROW_SAMPLE, clsslabel)
19
```

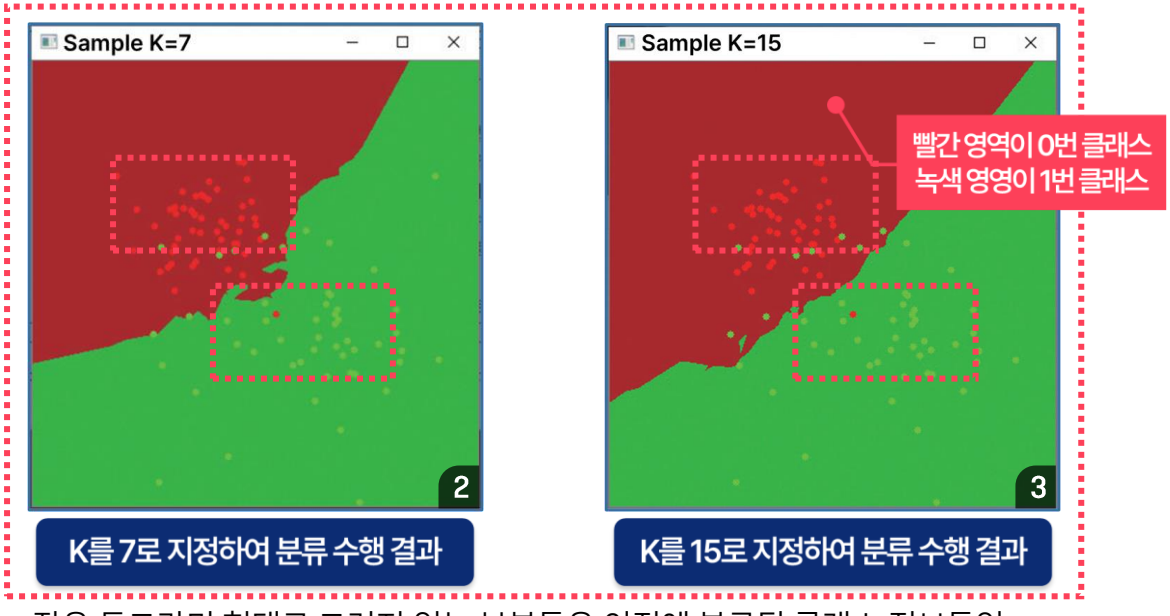
총 두 개의 그룹으로 총 100개의 데이터 생성

행렬원소에 정규분포를 따르는 임의 값 지정

두 그룹에 레이블 값 지정

가장 가까운 일곱 개의 값을 써서 분류하는 것을 의미

- 실행결과



→ 작은 동그라미 형태로 그려져 있는 부분들은 이전에 분류된 클래스 정보들임

→ 지정하고자 하는 픽셀의 위치는 빨간, 녹색 그룹에 해당되게 분류하여 정의

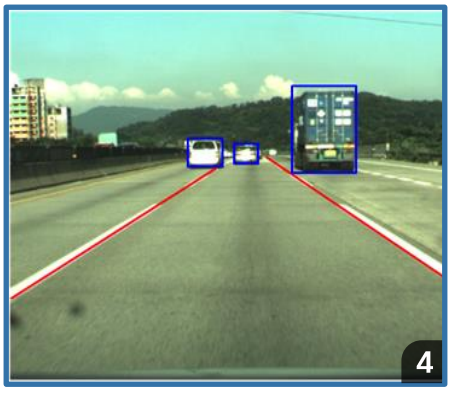
허프 변환(Hough Transform)

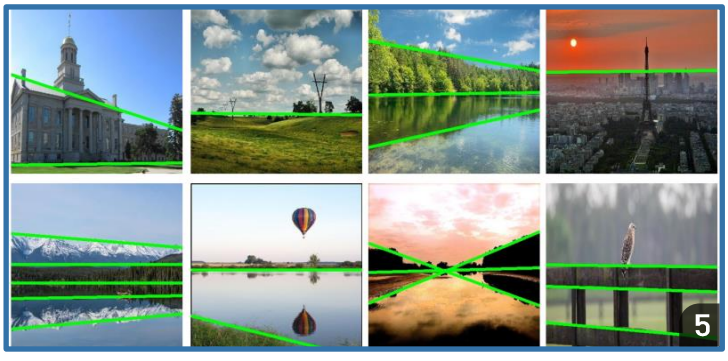
허프 변환(Hough Transform)

◆ 허프 변환(Hough Transform)

- 직선 검출

  → 차선 및 장애물 자동인식 시스템 - 차선 검출
  → 영상 내에서 공간 구조를 분석하는데 유용한 도구
  → 영상 처리와 컴퓨터 비전 분야에서 많은 연구 진행

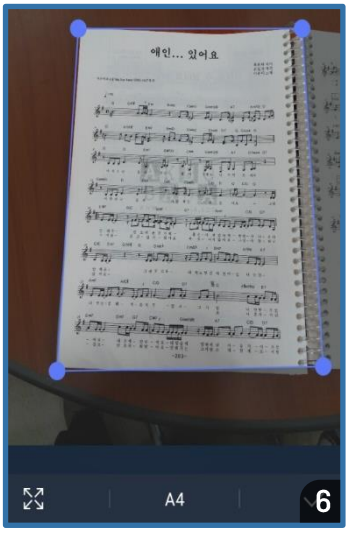




- 다양한 응용에 사용

  → 스캐너의 기능을 대신해 주는 앱 - 네 개 모서리 검출

  - Obtaining basic structure
  - Navigation/Track or lane assistants
  - Find vanishing points
  - Identifying objects by shape

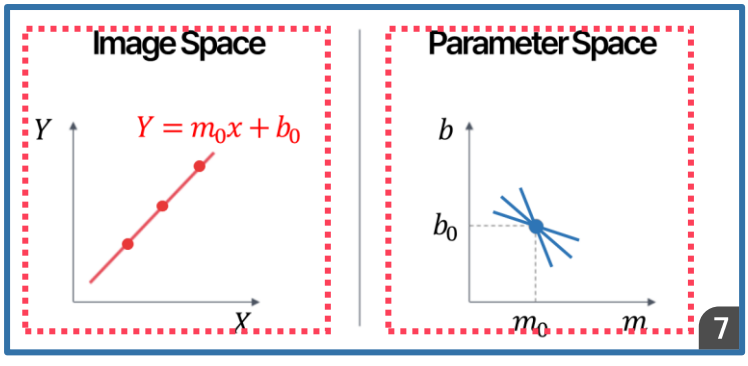


허프 변환(Hough Transform)

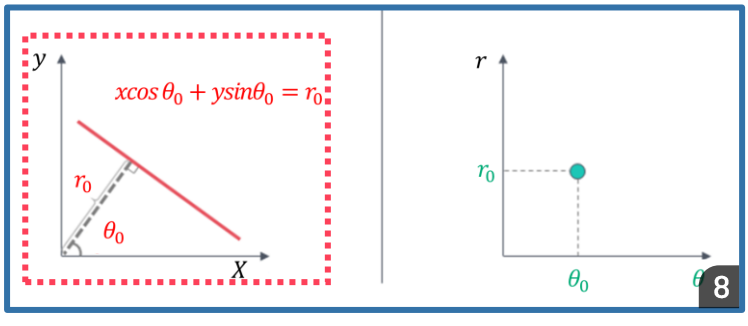
허프 변환(Hough Transform)

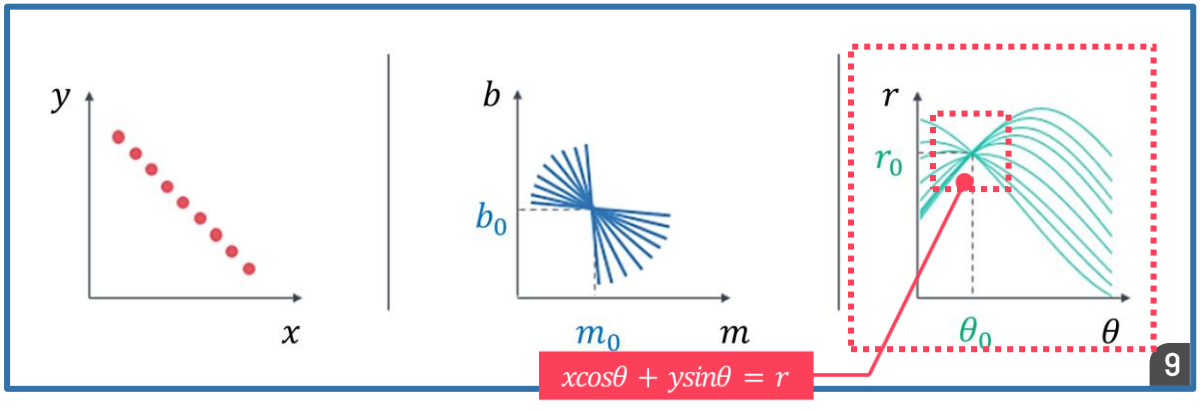
- 경계선 검출 영상 안에 존재하는 점 데이터를 이용해 차선 성분 검출
- 직선의 방정식

  → $y=mx+b$ >파라미터 m,b로 정의 가능



- r: 원점에서 직선까지의 거리
- θ: x축으로부터의 각도

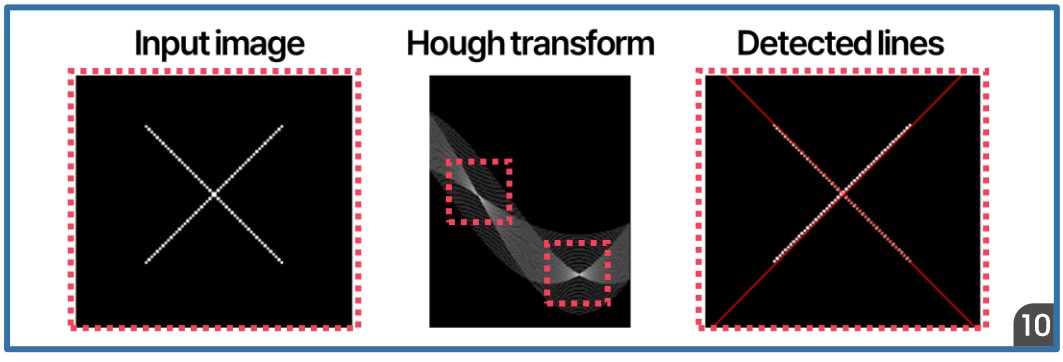




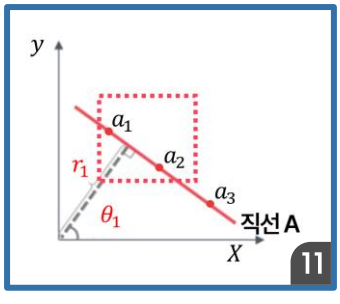
허프 변환(Hough Transform)

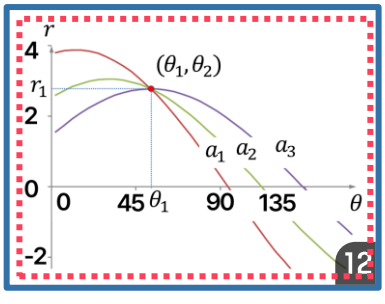
허프 변환(Hough Transform)

- 허프 공간의 r과 θ로 표현 > 공간 상의 하나로 점으로 수렴

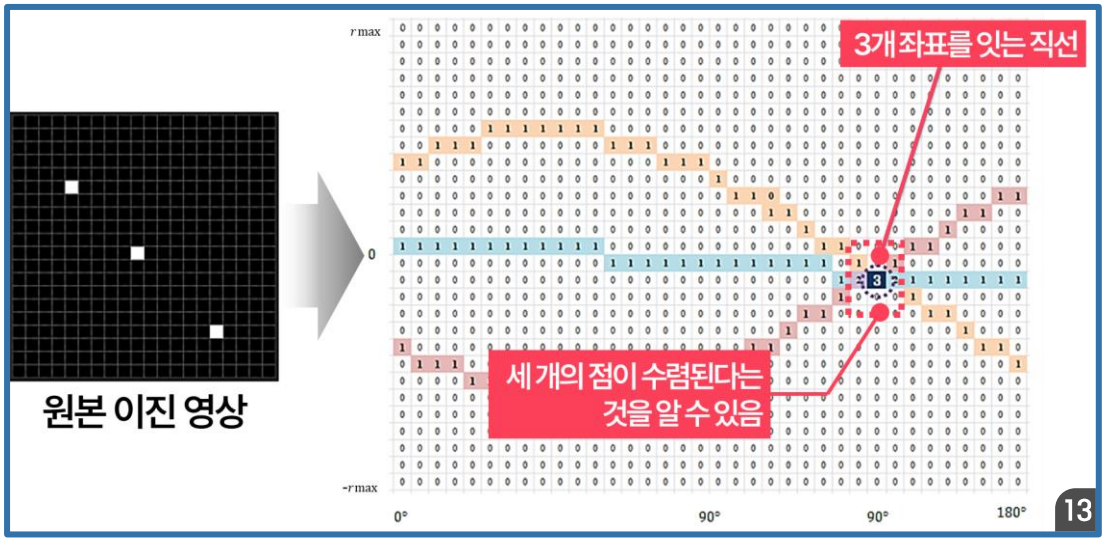


허프 변환(Hough Transform)의 좌표계





허프 누적 행렬 구성

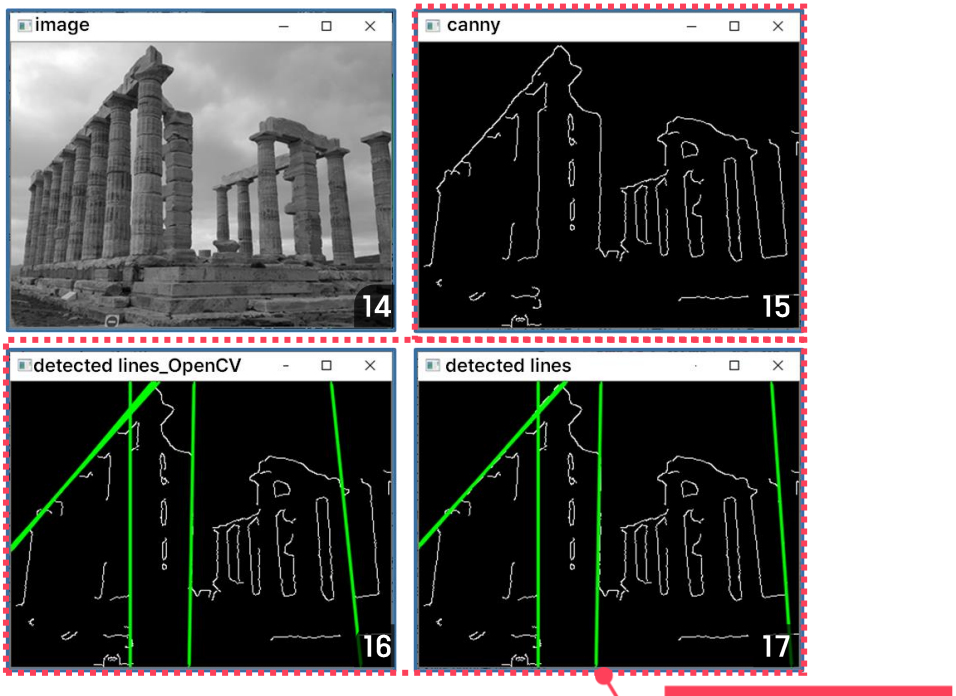


허프 변환(Hough Transform)

최종 완성 프로그램

```
25  image = cv2.imread("images/hough/.jpg", cv2.IMRED_GRAYSCALE)
26  if image is None: raise Excpetion("영상파일 읽기 에러")
27  blur = cv2.GaussianBlur(image, (5, 5), 2, 2)
28  canny = cv2.canny(blur, 100, 200, 5)
29
30  roh, theta = 1, np.pi / 180
31  lines1 =houghLines(canny, rho, theta, 80)
32  lines2 =cv2.HoughLines(canny, rho, theta, 80)
33  dst1 + draw_ houghLines(canny, lines, 7)
34  dst2 + draw_ houghLines(canny, lines, 7)
35
36  cv2.inshow("image", image)
37  cv2.inshow("canny", canny)
38  cv2.inshow("detected lines", dst1)
39  cv2.inshow(detected line_OpenCV", dst2)
40  cv2.waitkey(0)
```

- 실행결과

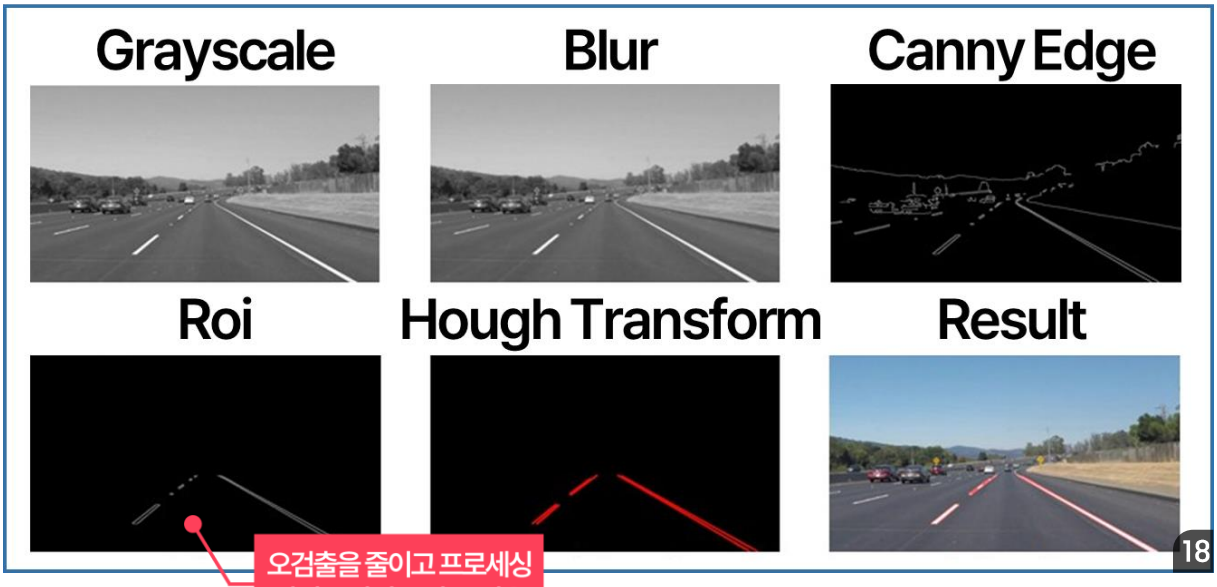


영상 처리 응용 사례

영상처리응용사례

영상 처리 응용 사례

- 하프 변환을 이용한 차선 검출
  - → 주행 중 도로상에 존재하는 차선을 검출
  - → 자율 주행 조향 제어를 위해 사용



하르 분류기를 이용한 얼굴 검출 및 성별 분류

- 얼굴 검출 및 인식 응용
  - → 저가형 디지털 카메라에서도 얼굴 검출 후 액정에 표시
  - → 페이스북이나 구글에 사진을 올리면 얼굴 영역 검출

정리하기

- 영상 분할 및 특징 추출
- K-최근접 이웃 분류기
- 허프 변환(Hough Transform)
- 영상 처리 응용 사례