패턴인식 실습 #8

Mean shift

리뷰) 과제 #5 – SIFT&SURF 속도 비교

- 임의의 이미지 한 장에 대하여 크기를 축소 및 확대시키면서 SIFT와 SURF의 keypoints 검출 속도를 각각 측정 후 출력하는 코드를 작성하세요. (단위: 초)
- .py 파일만 제출
- 예시)

```
>> (262, 336)
SIFT: 0.023991 sec
SURF: 0.198400 sec

>> (523, 672)
SIFT: 0.092752 sec
SURF: 0.053856 sec

>> (1046, 1344)
SIFT: 0.332113 sec
SURF: 0.209440 sec

>> (5230, 6720)
SIFT: 7.876117 sec
SURF: 3.709044 sec
```

파이썬 시간 측정 함수

- time 패키지
  - time 함수
    - 1970년 1월 1일 0시 0분 0초 이후 경과한 시간을 초 단위로 반환함
    - 시간대는 UTC(Universal Time Coordinated; 협정 세계시) 사용

  - 필요에 따라 선택하여 사용
    - perf_counter 함수
    - process_time 함수
    - 등등

```python
import cv2
import time

sift = cv2.xfeatures2d.SIFT_create()  # SIFT 검출기 생성
surf = cv2.xfeatures2d.SURF_create()  # SURF 검출기 생성

for scale_factor in [0.5, 1.0, 2.0, 10]:
    # 이미지 불러와서 리사이징
    img = cv2.imread('butterfly.png')
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    gray = cv2.resize(gray, None, fx=scale_factor, fy=scale_factor)
    print('>>', gray.shape)

    # SIFT 특징 검출 속도 계산
    t1 = time.time()
    keypoints = sift.detect(image=gray, mask=None)
    t2 = time.time()
    print('SIFT: %f sec' % (t2 - t1))

    # SURF 특징 검출 속도 계산
    t1 = time.time()
    keypoints = surf.detect(image=gray, mask=None)
    t2 = time.time()
    print('SURF: %f sec' % (t2 - t1))
```

> 출력 결과

```
>> (262, 336)
SIFT: 0.036901 sec
SURF: 0.214427 sec

>> (523, 672)
SIFT: 0.077791 sec
SURF: 0.044848 sec

>> (1046, 1344)
SIFT: 0.324134 sec
SURF: 0.184505 sec

>> (5230, 6720)
SIFT: 9.687593 sec
SURF: 3.765651 sec
```

개념 정리

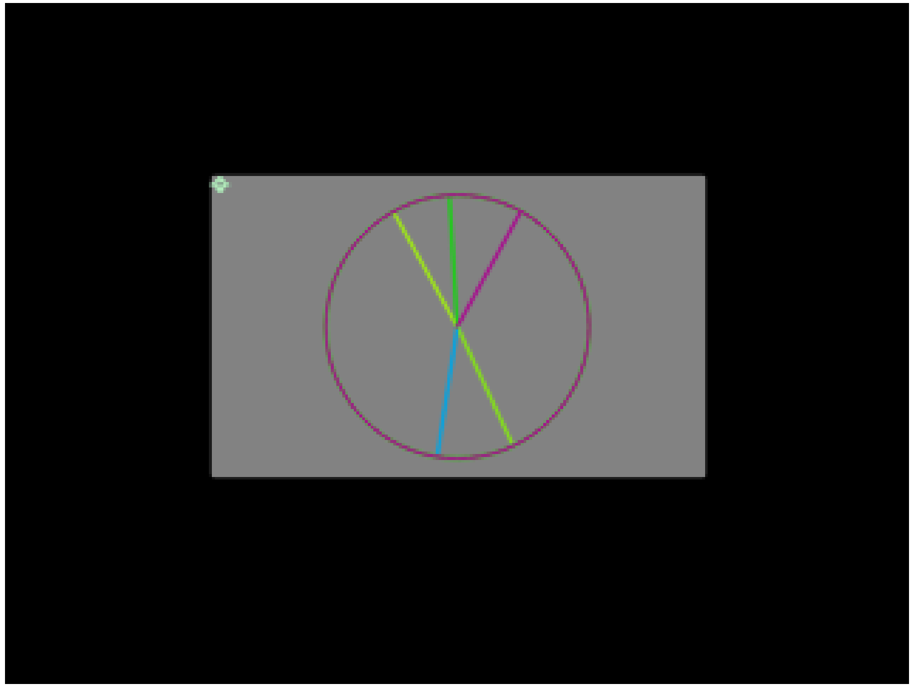
- 특징 추출기 (feature extractor): 특징의 위치를 찾는 기능
  - ‘4장 – 지역 특징 검출'에서 다룸

- 특징 기술자 (feature descriptor): 특징을 일련의 숫자(벡터)로 표현
  - ‘6장 – 특징 기술'에서 다룰 예정
  - 특징 기술자를 통해 생성된 특징 벡터 간 거리(distance) 또는 유사도(similarity) 계산

- 특징 매칭: 어떤 대상을 다른 것과 비교하여 같은 것인지 판단
  - ‘7장 – 매칭'에서 다룰 예정

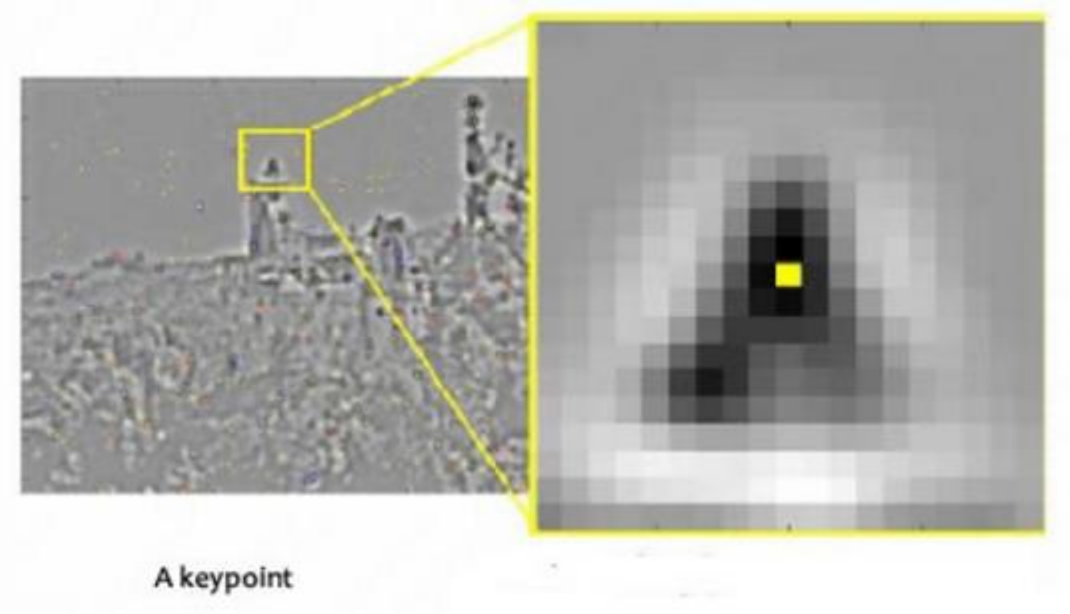
보충 설명

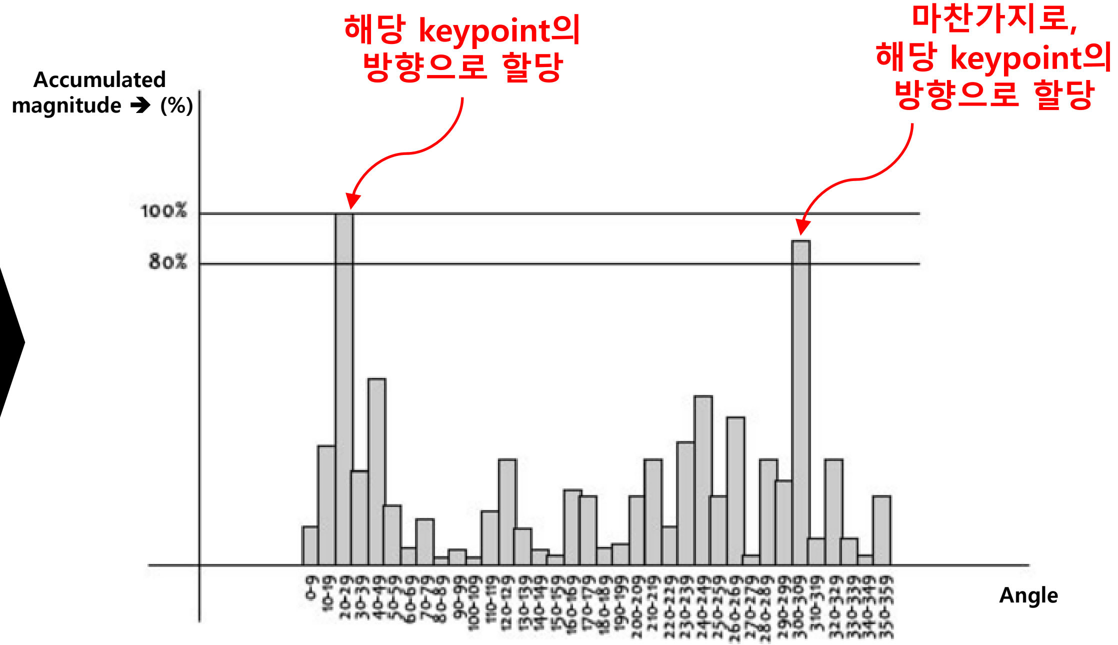
- 하나의 keypoint의 방향이 왜 여러 개 존재하는가?



④ keypoint에 방향(orientation) 할당

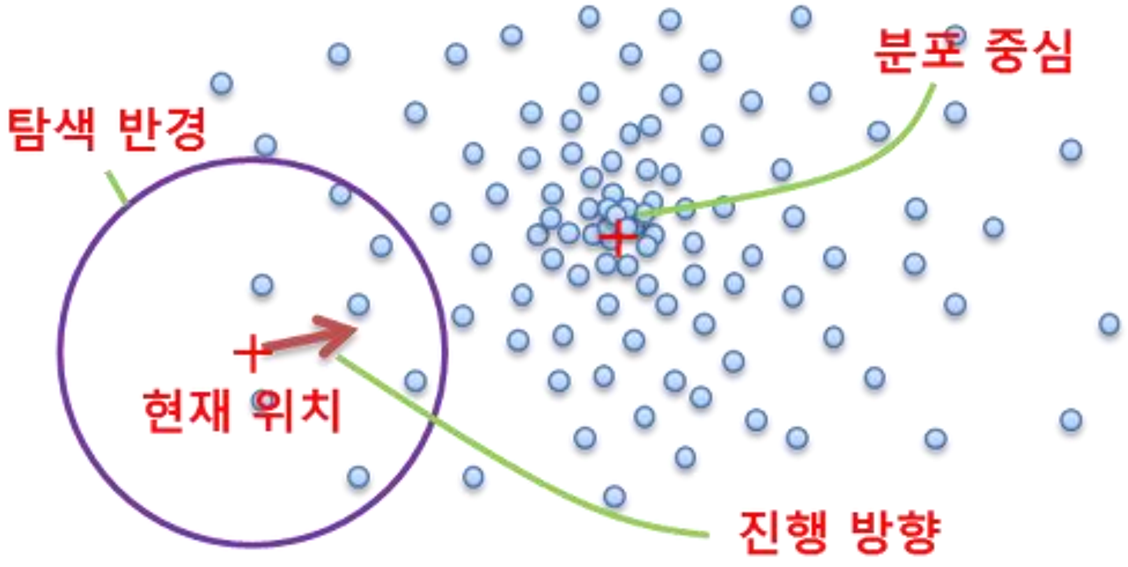
- 검출된 각 keypoint에 일정 크기의 윈도우를 씌운 후, 픽셀들의 gradient 방향 히스토그램 생성



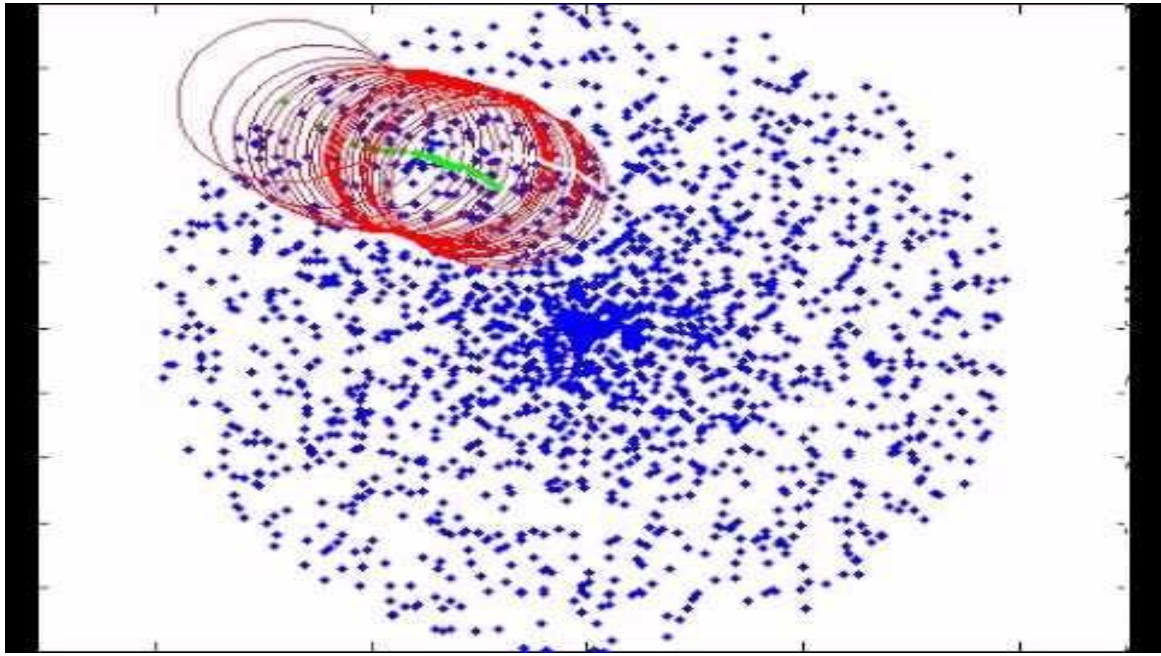


➔ 지배적인 방향을 찾아 해당 keypoint의 방향으로 할당

Mean shift 알고리즘



https://darkpgmr.tistory.com/64



https://youtu.be/hJg7ik4x95U

Mean shift 알고리즘의 특징

- k-means나 Gaussian mixture 군집화 알고리즘과 달리 군집의 개수를 사전에 알려줄 필요가 없음
- k-means나 Gaussian mixture는 일정한 모양의 분포를 가정하고 매개변수를 추정하는 모수적(parametric) 방법이지만, meanshift는 비모수적(nonparametric) 방법
- 설정해야 할 파라미터가 적음

- 단, 샘플 데이터 개수가 커질수록 속도가 느려짐
  - 샘플 데이터 수가 적고 군집의 개수를 사전에 알 수 없는 경우에 적합

Mean shift 알고리즘

알고리즘 5-6 민시프트를 이용한 군집화

입력 : 샘플 집합 $X=\{\mathbf{x}_i | i=1,2,\cdots,n\}$, ε(수렴 임계값)

출력 : k개의 모드 $\mathbf{z}_j$, $1 \le j \le k$, $\mathbf{x}_i$의 소속을 나타내는 $\alpha(\mathbf{x}_i)$, $1 \le i \le n$

```
for(i=1 to n) {
  y0=xi;  // 초기점 설정
  t=0;
  while(TRUE) {
    식 (5.19)를 이용하여 y_{t+1}을 계산한다.
    if(||y_{t+1} - y_t|| ≤ ε) break;  // 수렴
    t++;
  }
  v_i=y_{t+1};  // x_i의 수렴점을 저장
}

// 군집화 단계
v_i, i=1, 2, ···, n에서 h 이내에 있는 점들을 모아 군집화하고, 군집 중심을 z_j, j=1, 2, ···, k라 한다.
x_i, i=1, 2, ···, n이 속한 군집 z_c를 찾아 α(x_i)=c라 한다.
```

X_i: 입력 데이터의 i번째 샘플

Y_t: 시점 t에서의 군집 중심점

※ while 반복문 종료 조건

① 중심점 이동이 수렴한 경우

② t가 최대 반복 횟수를 초과한 경우

유클리디안(Euclidean) 거리 사용

V_i: i번째 샘플에 대한 최종 중심점

h : bandwidth (탐색 반경)

$$\text{평편한 커널}: k(\mathbf{x}) = \begin{cases} 1, & \|\mathbf{x}\| \le 1 \\ 0, & \|\mathbf{x}\| > 1 \end{cases}$$

$$\text{가우시안 커널}: k(\mathbf{x}) = \begin{cases} e^{-\|\mathbf{x}\|^2}, & \|\mathbf{x}\| \le 1 \\ 0, & \|\mathbf{x}\| > 1 \end{cases} \qquad (5.17)$$

- **y**를 $\mathbf{y}_0$로 놓고 시작하여, 수렴할 때까지 $\mathbf{y}_0 \rightarrow \mathbf{y}_1 \rightarrow \mathbf{y}_2 \rightarrow \ldots$ 반복

$$\mathbf{y}_{t+1} = \frac{\sum_{i=1}^{n} \mathbf{x}_i k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^{n} k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)} \qquad (5.18)$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \mathbf{m}(\mathbf{y}_t)$$

이때 $\mathbf{m}(\mathbf{y}_t) = \mathbf{y}_{t+1} - \mathbf{y}_t$

$$= \frac{\sum_{i=1}^{n} \mathbf{x}_i k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^{n} k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)} - \mathbf{y}_t \qquad (5.19)$$

$$= \frac{\sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{y}_t) k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^{n} k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}$$

Mean shift 알고리즘

알고리즘 5-6 민시프트를 이용한 군집화

입력 : 샘플 집합 $X=\{\mathbf{x}_i | i=1,2,\cdots,n\}$, ε(수렴 임계값)

출력 : k개의 모드 $\mathbf{z}_j$, $1 \le j \le k$, $\mathbf{x}_i$의 소속을 나타내는 $\alpha(\mathbf{x}_i)$, $1 \le i \le n$

```
for(i=1 to n) {
  y0=xi;  // 초기점 설정
  t=0;
  while(TRUE) {
    식 (5.19)를 이용하여 y(t+1)을 계산한다.
    if(||y(t+1) - y(t)|| ≤ ε) break;  // 수렴
    t++;
  }
  vi=y(t+1);  // xi의 수렴점을 저장
}

// 군집화 단계
vi, i=1, 2, …, n에서 h 이내에 있는 점들을 모아 군집화하고, 군집 중심을 zj, j=1, 2, …, k라 한다.
xi, i=1, 2, …, n이 속한 군집 zc를 찾아 α(xi)=c라 한다.
```

X_i: 입력 데이터의 i번째 샘플

Y_t: 시점 t에서의 군집 중심점

※ while 반복문 종료 조건

① 중심점 이동이 수렴한 경우

② t가 최대 반복 횟수를 초과한 경우

과제에서 완성할 부분!

유클리디안(Euclidean) 거리 사용

V_i: i번째 샘플에 대한 최종 중심점

h : bandwidth (탐색 반경)

$$\text{평편한 커널}: k(\mathbf{x}) = \begin{cases} 1, & \|\mathbf{x}\| \le 1 \\ 0, & \|\mathbf{x}\| > 1 \end{cases}$$
$$\text{가우시안 커널}: k(\mathbf{x}) = \begin{cases} e^{-\|\mathbf{x}\|^2}, & \|\mathbf{x}\| \le 1 \\ 0, & \|\mathbf{x}\| > 1 \end{cases} \tag{5.17}$$

- **y**를 $\mathbf{y}_0$로 놓고 시작하여, 수렴할 때까지 $\mathbf{y}_0 \rightarrow \mathbf{y}_1 \rightarrow \mathbf{y}_2 \rightarrow \ldots$ 반복

$$\mathbf{y}_{t+1} = \frac{\sum_{i=1}^{n} \mathbf{x}_i k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^{n} k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)} \tag{5.18}$$

$$\mathbf{y}_{t+1} = \mathbf{y}_t + \mathbf{m}(\mathbf{y}_t)$$

이때 $\mathbf{m}(\mathbf{y}_t) = \mathbf{y}_{t+1} - \mathbf{y}_t$

$$= \frac{\sum_{i=1}^{n} \mathbf{x}_i k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^{n} k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)} - \mathbf{y}_t$$
$$= \frac{\sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{y}_t) k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)}{\sum_{i=1}^{n} k\left(\frac{\mathbf{x}_i - \mathbf{y}_t}{h}\right)} \tag{5.19}$$

실습 환경 세팅

- Jupyter lab 설치
  - pip install jupyterlab
- Jupyter kernel에 가상환경 추가
  - python –m ipykernel install –user –name=*env_name*
- Jupyter lab 실행
  - (optional) activate *env_name*
  - jupyter lab

과제 #6 – Mean shift 함수 완성하기

- 교재 슬라이드에 주어진 수식을 참고하여 meanshift.py 파일에 있는 mean_shift 함수를 완성하세요.
(<u>.py 파일만 제출</u>)