MeanShift를 통한 객체추적

7.1 역투영

히스토그램

- 히스토그램은 데이터의 분포를 나타내는 그래프이며, 이미지의 히스토그램은 픽셀 값(또는 특정 채널 값: Hue, Saturation 등)의 빈도를 나타낸다.
- 예를 들어, HSV 색 공간에서 Hue(색상)와 Saturation(채도)의 조합으로 히스토그램을 계산할 수 있음

히스토그램 역투영(backprojection)

- 색상 히스토그램을 사용해 이미지 데이터베이스에서 유사한 이미지를 빠르게 검색하고 인덱싱하는 방법
- 히스토그램 역투영은 영상의 각 픽셀이 주어진 히스토그램 모델에 얼마나 일치하는지를 검사하는 방법
- 임의의 색상 영역을 검출할 때 효과적
- 역투영은 특정 히스토그램 모델(ROI의 히스토그램)을 기준으로 전체 이미지의 픽셀을 비교하고, 모델과 유사한 픽셀의 확률을 계산하여 결과를 반환하는 과정
- 결과 이미지는 원본 이미지와 동일한 크기를 가지며, 각 픽셀의 값은 해당 픽셀이 ROI 히스토그램과 얼마나 일치하는지를 나타냄

7.1 역투영

관심영역(ROI)

관심영역(ROI)과 유사한 색상을 갖는 화소를 추출한 결과

mask 영상을 이용하여
원본 이미지에서 색상 추출

7.1 역투영

히스토그램 역투영의 단계

- 관심 영역(ROI) 설정 및 히스토그램 계산
  1) 분석 대상 이미지에서 관심 영역(ROI)을 선택
  2) ROI의 히스토그램을 계산
    - 색상 기반 분석에서는 조명변화에 덜 민감한 HSV 색 공간을 주로 사용하며, Hue(색상)와 Saturation(채도)를 기준으로 2D 히스토그램을 계산
    - cv2.calcHist를 사용하여 히스토그램을 계산, cv2.normalize를 통해 정규화(normalization)
- 히스토그램 역투영 수행
  1) 검색 대상 이미지의 각 픽셀을 ROI 히스토그램과 비교
  2) 각 픽셀이 ROI 히스토그램에 얼마나 일치하는지 계산하여 "유사도"를 결정
    - ROI 히스토그램에서의 확률 값을 사용하여 각 픽셀의 유사도를 계산합니다.
  3) 계산 결과는 픽셀 강도(0~255)로 표현됩니다.
- 후처리
  1) 역투영 결과를 이진화하여 관심 영역만 강조할 수 있습니다.
    - cv2.threshold를 사용하여 특정 임계값 이상인 부분만 유지합니다.
  2) 마스크를 사용하여 원본 이미지에 강조된 결과를 표시하거나 특정 영역을 추적합니다.

7.1 역투영

히스토그램 역투영 예제

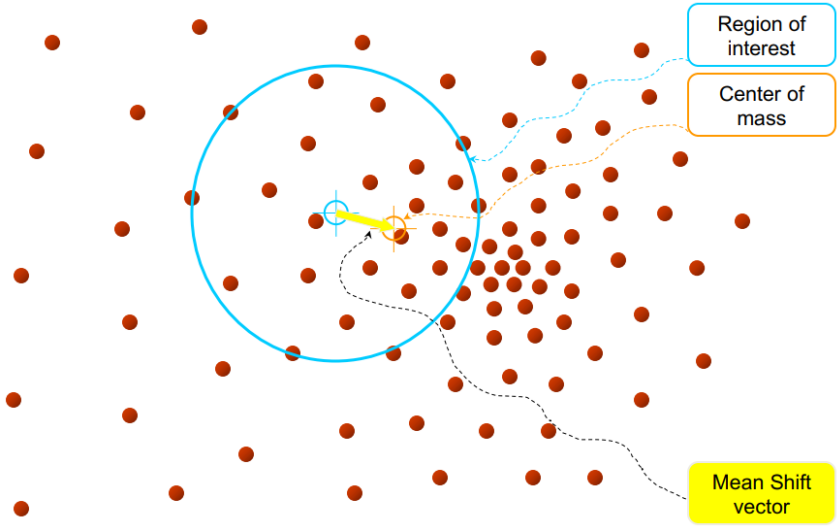
- cv2.calcBackProjection 함수를 이용하여 히스토그램 역투영

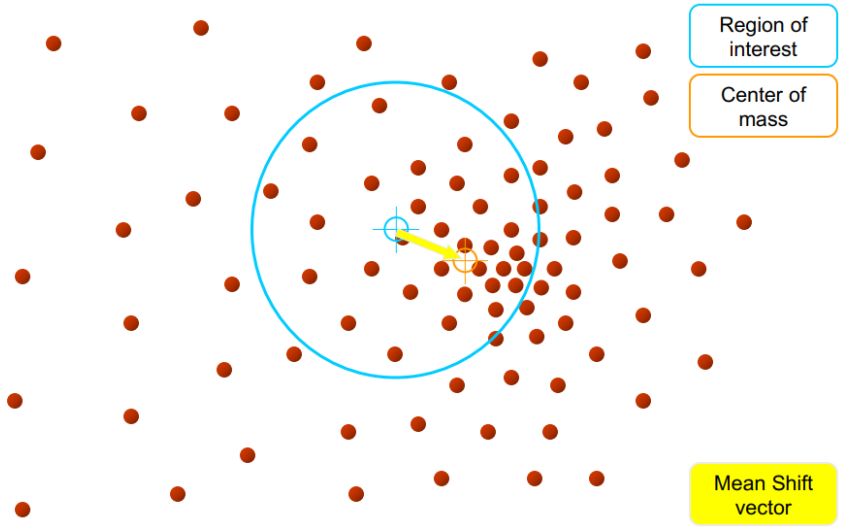
- cv2.calcBackProjection(images, channels, hist, ranges, scale, dst=None) -> dst
  - images: 입력 영상 리스트
  - channels: 역투영 계산에 사용할 채널 번호 리스트
  - hist: 입력 히스토그램 (numpy.ndarray)
  - ranges: 히스토그램 각 차원의 최솟값과 최댓값으로 구성된 리스트
  - scale: 출력 역투영 행렬에 추가적으로 곱할 값
  - dst: 출력 역투영 영상. 입력 영상과 동일 크기, cv2.CV_8U.

7.2 MeanShift

Meanshift란

- Mean Shift는 비지도 학습의 한 종류로, 주어진 데이터의 밀도가 높은 영역(모드)을 찾아가는 모드 탐지 알고리즘
- 데이터를 반복적으로 이동하면서 데이터의 밀도가 높은 지역(모드)로 수렴
- 클러스터링(데이터 군집화)과 객체 추적 등 다양한 응용 분야에서 사용





7.2 MeanShift

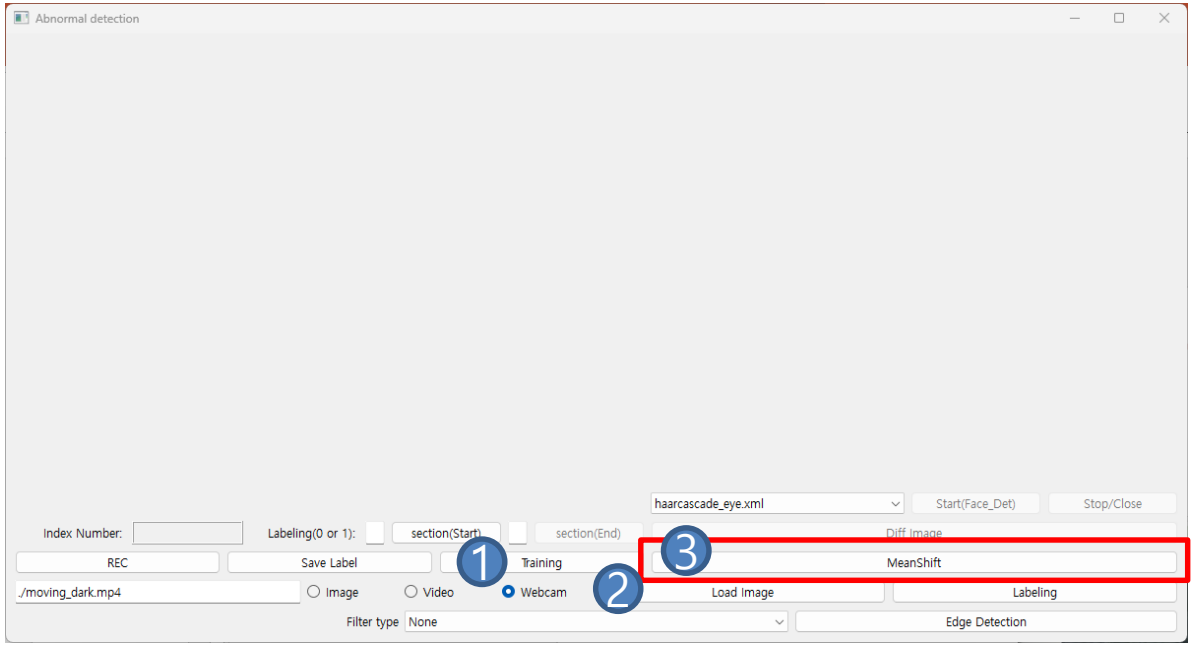
- Meanshift의 특징
  - 장점
    - 클러스터 개수를 미리 지정할 필요 없음(자동 탐지)
    - 데이터 분포를 이해하는 데 유용
    - 색상 기반으로 빠르고 간단한 객체 추적 가능
    - 실시간 처리에 적합
    - 초기 설정만 잘 하면 성능이 우수
  - 단점
    - 배경과 객체의 색상이 유사하면 추적 실패 가능
    - 크기 변화, 회전, 조명 변화에 민감
    - 빠르게 이동하는 객체를 추적하기 어려움
    - 대역폭 h설정이 매우 중요. h가 작으면 과적합(Noise 포함) 가능
    - h가 크면 과소적합(모드가 정확하지 않음) 가능
    - 계산 비용이 높아 대규모 데이터에 적용하기 어려움

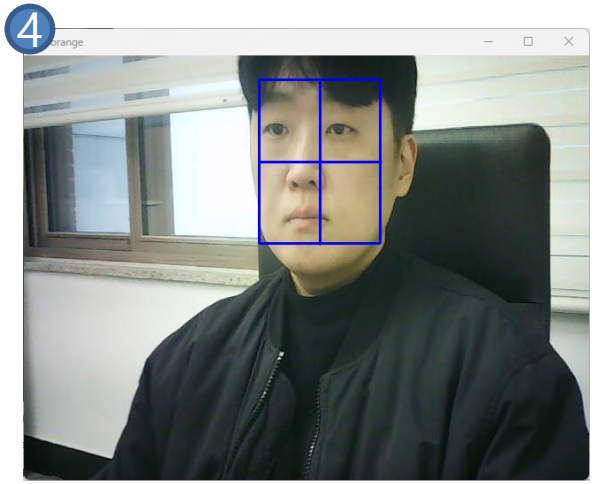
7.2 MeanShift

- Meanshift를 이용한 객체 추적의 단계
  - 초기 설정
    - 첫 번째 프레임에서 추적할 객체의 ROI를 설정
    - ROI의 색상 히스토그램을 계산하여, 객체를 표현하는 특징 모델로 사용
  - 히스토그램 역투영
    - 새로운 프레임에서, 각 픽셀의 색상 값을 ROI의 히스토그램과 비교하여 유사도를 계산
    - 결과는 관심 영역과 유사한 픽셀을 강조하는 흑백 이미지로 나타남
  - Mean Shift 알고리즘 적용:
    - 검색 창(Tracking Window)의 중심을 시작점으로 설정.
    - 검색 창 내부에서 히스토그램 값이 높은 픽셀을 기준으로 새로운 중심을 계산(Mean Shift 벡터).
    - 중심 이동을 반복하여 모드(밀도가 가장 높은 지점)에 수렴.
  - 추적 창 갱신:
    - 새로운 중심 좌표를 기준으로 검색 창의 위치를 갱신.
    - 추적 창은 다음 프레임으로 전달됩니다.

7.3 meanshift를 통한 화면 전환 구현

- 1) webcam 라디오 버튼 선택
- 2) load image 버튼 클릭
- 3) meanshift 버튼 클릭
- 4) roi 영역 드래그로 그리기 후 엔터





7.3 meanshift를 통한 화면 전환 구현

- 구현 및 실습
  - (필수) 기존 코드에 다음 요소들 추가하기
    - "meanshift" 버튼 추가
  - (필수) Thread_out 클래스의 run 함수 안에 meanshift 동작 코드 삽입
  - (필수) meanshift 박스 중심에 빨강색 점 찍기
  - (필수) 이전 시간의 mouse 2개 점 이벤트에 의한 ROI 사각형 영역에 meanshift 중심점이 들어갈 경우 화면 전환 구현