Computer Vision and Deep Learning

Ch 05

강명묵

Contents

Matching

Matching : 물체 인식, 물체 추적, 스테레오, 카메라 캘리브레이션 등에 사용

$$A = \{a_1, a_2, \cdots, a_m\}$$
$$B = \{b_1, b_2, \cdots, b_n\}$$

여기에 해당하는 a_i와 b_j쌍을 모두 찾는 문제

두 기술자 간 거리 측정 방식으로 **Euclidean distance** 방식 사용

$$d(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{k=1,d} (a_k^2 - b_k^2)^2} = ||a - b|| \qquad (3.1)$$

Matching

매칭 전략

매칭 전략 1 _ 고정 임계값

$$d(a_i, b_j) < T \qquad (3.2)$$

- T가 너무 작을 때
  - 매칭 쌍이 조건을 통과하지 못해 false negative 많이 발생
- T가 너무 클 때
  - 조금만 비슷해도 조건을 만족하므로 false positive 많이 발생

매칭 전략 2 _ 최근접 이웃 거리 비율

$$\frac{d(a_i, b_j)}{d(a_i, b_k)} < T \qquad (3.3)$$

- a_i는 B에서 가장 가까운 b_j와 두 번째 가까운 b_k를 찾는다.
- b_j 와 b_k가 식 (3.3)을 만족하면 a_i와 b_j는 매칭 쌍이 된다.

Matching

성능 측정

혼동행렬

		정답(GT)	
		긍정	부정
예측	긍정	참 긍정(TP)	거짓 긍정(FP)
	부정	거짓 부정(FN)	참 부정(TN)

$$precesion = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

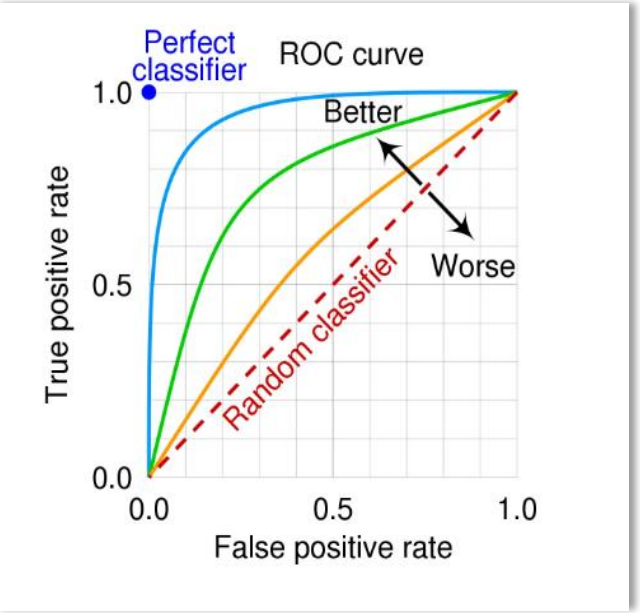
$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Matching

성능 측정

ROC curve와 AUC



- T가 작아지면 FPR(False Positive Rate)는 작아짐
- T가 커지면 FPR이 커지는데 TPR(True Positive Rate)도 따라 커진다.

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{TN + FP}$$

Matching

빠른 매칭

1. kd tree

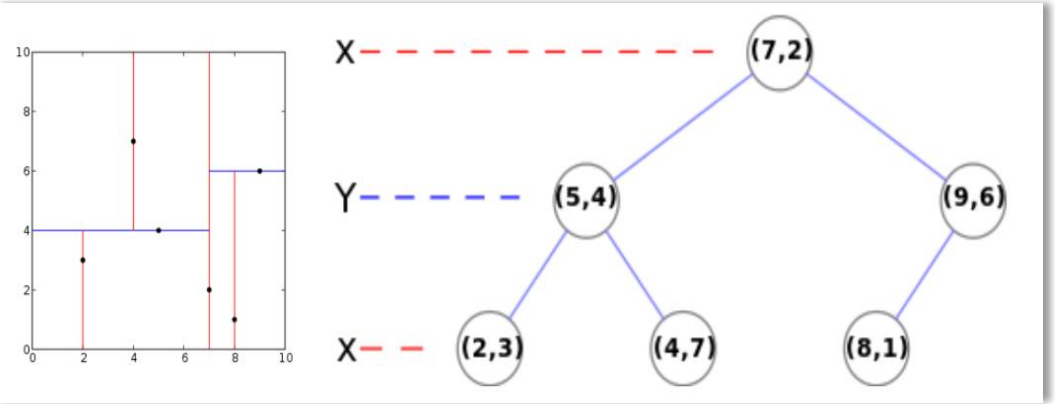
- 기술자 집합 $\mathbf{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_m\}$, i번째 기술자 $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \cdots, x_{id})$

1. Loot node 생성
   - d개의 축 각 분산 계산 후, **분산이 가장 큰 축**(k) 선택

2. **X**를 두 부분집합으로 분할($\boldsymbol{X}_{left}, \boldsymbol{X}_{right}$)

3. 중앙값 결정(x_{jk})
   - k번째 요소 $(x_{1k}, x_{2k}, \cdots, x_{mk})$ 정렬 후 중앙값 결정

4. 루트 노드에 x_j 배치 후 $\boldsymbol{X}_{left}, \boldsymbol{X}_{right}$로 분할
   - k번째 축의 값이 x_{jk}보다 작으면 왼쪽
   - k번째 축의 값이 x_{jk}보다 크면 오른쪽
   - 해당 과정 **재귀적**으로 실행



관련 라이브러리 : **FLANN**

Matching

빠른 매칭

2. Locality-Sensitive Hashing (LSH)

- 데이터의 위치나 공간적인 특성을 고려하여 해싱을 수행하는 기술
- **Jaccard Similarity**가 높은 원소들을 같은 버킷에 넣는 해쉬 알고리즘

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Algorithm

1. 해싱 함수 정의
2. 벡터 양자화
3. 클러스터링
4. 해시 테이블 구축
5. 유사성 검색

관련 라이브러리 : **FAISS(Facebok AI Similaraity Search)**

Matching

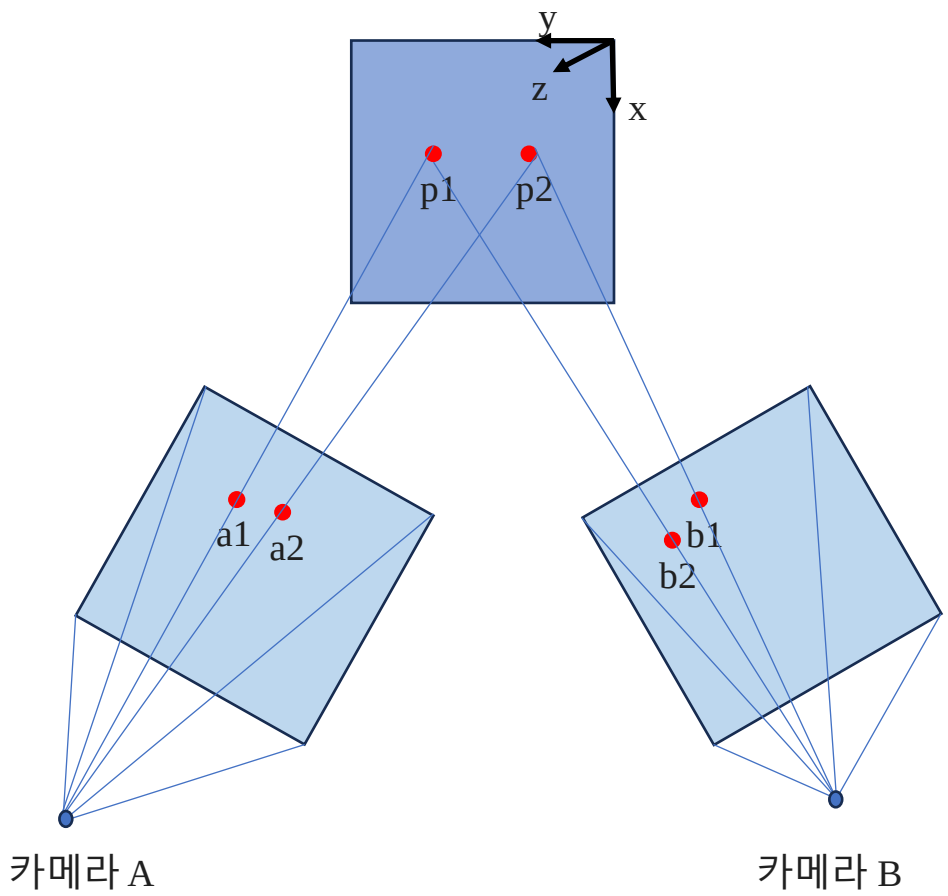
호모그래피

호모그래피 추정

- Outlier를 걸러내기 위함
- 매칭 쌍을 이용하여 물체 위치를 찾을 수 있게 해줌

$$\mathbf{b}^T = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & 1 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ 1 \end{bmatrix} = \mathbf{H}\mathbf{a}^T$$

매칭 쌍 n개 $(a_1, b_1), (a_2, b_2), (a_3, b_3), \cdots, (a_n, b_n)$



Matching

호모그래피

강인한 호모그래피 추정

매칭 쌍 n개 $(a_1, b_1), (a_2, b_2), (a_3, b_3), \cdots, (a_n, b_n)$

- B는 b_i를 i번째 열에 배치한 3xn 행렬
- A는 a_i를 i번째 열에 배치한 3xn 행렬
- B = HA

LMSE(Least Mean Square Error)

$$E = \frac{1}{n} \sum_{i=1,n} ||\mathbf{H}\mathbf{a}_i^T - \mathbf{b}_i||_2^2$$

- 아웃라이어 걸러내는 방법
  - 평균 대신 **중앙값**(meadian)을 사용
    - n개의 오차를 계산하고 가운데 위치한 오차를 E로 취함
    - 하지만 더 강인한 추정기법(RANSAC) 존재

Matching

호모그래피

$$\frac{d(a_i, b_j)}{d(a_i, b_k)} < T \qquad (3.3)$$

RANSAC

1. 후보 호모그래피 저장할 리스트 **h** 초기화
2. 지정된 횟수 m만큼 반복 3~9
3. X에서 네 쌍을 랜덤하게 선택하고 호모그래피 행렬 **H** 추정
4. 이 쌍으로 inlier 집합 초기화
5. for 3번째에서 선택한 네 쌍을 제외한 모든 쌍 p에 대해
6. if 만약 p가 허용 오차 t 이내로 **H**에 적합하면 p를 inlier에 삽입
7. if inlier가 d개 이상의 요소를 가지면
8. inlier의 모든 요소를 가지고 호모그래피 행렬 **H** 다시 추정
9. if 8행에서 적합 오차가 e보다 작으면 **H**를 h에 삽입
10. h에 있는 호모그래피 중에서 가장 좋은 것을 $\hat{\mathbf{H}}$으로 취함.

PROSAC

1. 후보 호모그래피 저장할 리스트 **h** 초기화
2. 지정된 횟수 m만큼 반복 3~9
3. X에서 네 쌍을 **차별하여** 선택하고 호모그래피 행렬 **H** 추정
   - 식 (3.3)에서 설정한 임계값 T보다 더 작은 값으로 통과한 쌍일수록 뽑힐 확률을 높여주는 전략 사용
4. 이 쌍으로 inlier 집합 초기화
5. for 3번째에서 선택한 네 쌍을 제외한 모든 쌍 p에 대해
6. if 만약 p가 허용 오차 t 이내로 **H**에 적합하면 p를 inlier에 삽입
7. if inlier가 d개 이상의 요소를 가지면
8. inlier의 모든 요소를 가지고 호모그래피 행렬 **H** 다시 추정
9. if 8행에서 적합 오차가 e보다 작으면 **H**를 h에 삽입
10. h에 있는 호모그래피 중에서 가장 좋은 것을 $\hat{\mathbf{H}}$으로 취함.

입력 : 매칭 쌍 집합 X = $\{(a_1, b_1), (a_2, b_2), \cdots, (a_n, b_n)\}$, 반복 횟수 m, 임곗값 t, d, e

출력 : 최적 호모그래피 $\hat{\mathbf{H}}$

Matching

호모그래피 추정 실험

$$\frac{d(a_i, b_j)}{d(a_i, b_k)} < T \qquad (3.3)$$

감사합니다.