

Hue Moments

2024년 1학기

서경대학교 김진헌

Structural Analysis and Shape Descriptors

차례

1

- 1. 개요
- 2. 어떻게 연산하나?
 - ▣ 2.1 예제: Hu Moments 연산
- 3. Matching with Hu Moments
 - ▣ 3.1 Matching Score 연산식
 - ▣ 3.2 사례: 특정 컨투어와의 매칭 점수 비교
 - ▣ 3.3 매칭 점수에 대한 고찰
 - ▣ 3.4 회전과 반사에 대한 휴 모멘트 변화 관찰
- 4. 검토

1. 개요

2

chatGPT에 질의한 결과 정리

- Hu 모멘트 불변량(Hu moment invariant)은 무게 중심과 대칭성에 기반한 이미지 특징 추출 방법이다.
 - ▣ 1962년에 미국의 컴퓨터 과학자인 M.K. Hu가 개발하였다.
 - 이를 가능하게 하는 주요 아이디어는 이미지의 무게 중심과 대칭성에 관련된 모멘트를 사용하여 이미지의 특징을 설명하는 것이다.
 - ▣ 이미지의 형태와 구조에 대한 정보를 추출하기 때문에 회전, 크기 변화, 위치 이동, 반사 등에 불변하게 유지된다.
 - 이는 객체 인식, 패턴 인식, 이미지 매칭 등의 컴퓨터 비전 응용 프로그램에서 널리 사용된다.
- $hu[0] \sim hu[6]$ 까지 총 7개의 특징량이다.
 - 위치이동, 크기의 변화와 회전, 반사(reflection)에 영향받지 않는데 반사의 경우 [6]번은 부호가 반대로 된다.
 - 영상이 무한대의 해상도를 갖고 있다는 가정에서 불변함이 가정되기 때문에 비트맵을 사용하는 raster image에서는 조금 달라질 수 있다.

2. 어떻게 연산하나?

3

- 1) 일단 모멘트, m 을 구한다. ← 영상 혹은 컨투어(1개) 모멘트

- $m = cv.moments(img)$ 혹은 $m = cv.moments(contour)$

- 2) m 으로 부터 휴 모멘트를 구하는 함수를 호출한다.

- $Hu = cv.HuMoments(m[, hu])$

- m : $moments()$ 함수로 연산한 모멘트

- Hu : 선택적, 출력=반환 값

휴 모멘트는 모멘트 중에서
정규화 중앙 모멘트로
연산한다.

- 반환 값 : Hu 모멘트 값. 7개의 모멘트 값으로 이루어진 구조체

$$hu[0] = \eta_{20} + \eta_{02}$$

$$hu[1] = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$

$$hu[2] = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$hu[3] = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$hu[4] = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

$$hu[5] = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

$$hu[6] = (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

where η_{ji} stands for **Moments::nu_{ji}**

OpenCV에서는 'nu'라고 표기했는데
여기서는 그리스 문자 η (eta)로 표기되었다.

2.1 예제: Hu Moments 연산

4

4_hu_moments.py

- 객체가 1개 있는 영상에 대해 영상 모멘트와 컨투어 영상 모멘트를 구하고 이를 기반으로 Hu Moments를 구하여 출력해 본다.

step1: binary image moments:

```
{'m00': 27483.0, 'm10': 5215132.0, 'm01': 3710855.0,
```

1) binary image moments based Hu moments:

```
[[1.85435586e-01]  
[9.05552265e-03]  
[8.20582863e-09]  
[9.72290877e-10]  
[2.72217742e-18]  
[9.05712660e-11]  
[3.63560712e-19]]
```

Step2: external contour moments of 0th contour:

```
{'m00': 27197.0, 'm10': 5160919.0, 'm01': 3672240.6666
```

2) external contour based Hu moments:

```
[[1.85777225e-01]  
[9.18126160e-03]  
[8.39338126e-09]  
[1.00427678e-09]  
[2.88816357e-18]  
[9.40732890e-11]  
[4.00054444e-19]]
```

1. 영상 모멘트와 컨투어 모멘트 값이 서로 유사하다.
2. 각 모멘트를 기반으로 연산한 Hu moments 값들도 서로 유사하다.

3. Matching with Hu Moments

5

- OpenCV는 template 영상과 target 영상과의 각각의 7개의 휴 모멘트의 결과를 바탕으로 약간의 연산을 통해 그 매칭 점수 (matching score)를 연산해 내는 함수를 지원한다.
- `retval = cv.matchShapes(contour1, contour2, method, parameter)`
 - 영상 혹은 윤곽선 2개에 대한 유사성을 Hu 모멘트를 이용하여 계산하는 함수.
 - 입력을 컨투어를 받고 내부에서 Hu 모멘트를 계산하여 비교한 후 유사성을 스칼라 값으로 반환한다.
 - Compares two shapes. The function compares two shapes. All three implemented methods use the Hu invariants (see [HuMoments](#))

Parameters

contour1 First contour or grayscale image.

contour2 Second contour or grayscale image.

method Comparison method, see [ShapeMatchModes](#)

parameter Method-specific parameter (not supported now).


마지막 파라미터는 생략하면 오류 발생. 0.0이라도 넣어야 한다.

3.1 Matching Score 연산식

6

7개의 휴 모멘트의 값의 거리를 다음 3가지 방식으로 측정

Table 8-2. Matching methods used by cvMatchShapes()

Value of method	cvMatchShapes() return value	
CV_CONTOURS_MATCH_I1	$I_1(A,B) = \sum_{i=1}^7 \left \frac{1}{m_i^A} - \frac{1}{m_i^B} \right $	두 영상이 같으면 세 지표 모두가 0이 됨
CV_CONTOURS_MATCH_I2	$I_2(A,B) = \sum_{i=1}^7 m_i^A - m_i^B $	
CV_CONTOURS_MATCH_I3	$I_3(A,B) = \sum_{i=1}^7 \left \frac{m_i^A - m_i^B}{m_i^A} \right $	
$m_i^A = \text{sign}(h_i^A) \cdot \log h_i^A $ $m_i^B = \text{sign}(h_i^B) \cdot \log h_i^B $		<div>  휴 모멘트의 크기에 대한 log 스케일링, 부호는 보존 </div>

where h_i^A and h_i^B are the Hu moments of A and B , respectively.

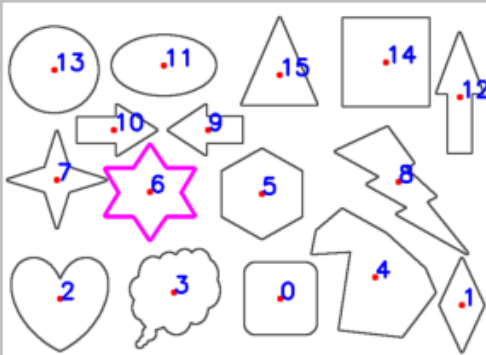
3.2 사례: 특정 컨투어와의 매칭 점수 비교

7

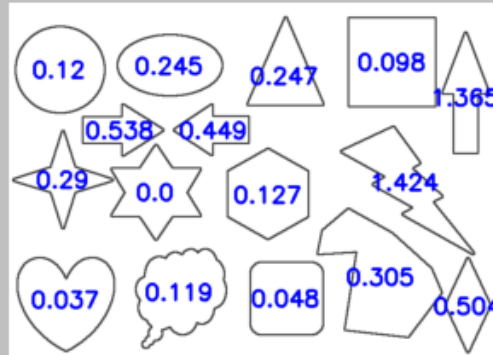
5_matching_every_shape.py

Total number of shapes found=16
Contour number to be matched=6

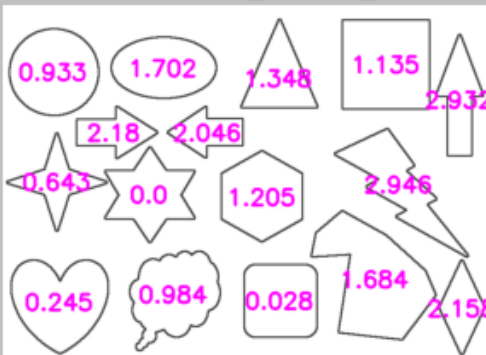
template contour number=6



CONTOURS_MATCH_I1



CONTOURS_MATCH_I2



CONTOURS_MATCH_I3



입력 영상:

'match_shapes3.png'

이진화 과정을 거친 영상에 대해 검출한 컨투어 객체에 대해 랜덤 함수로 선택한 컨투어를 번호를 기준 (template)으로 할 때 각 객체와의 Hu moment의 매칭 score를 3가지 관점(I1~I3)으로 보였다.

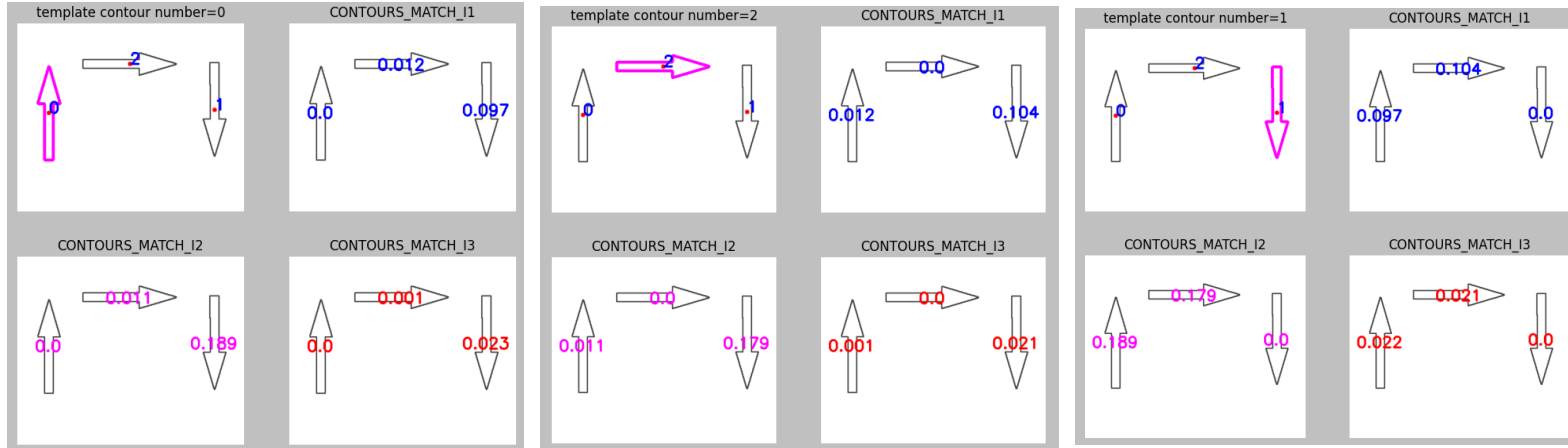
그림에서 좌측 상단은 검출한 컨투어와 그 번호를 보였고, 임의의 선택된 객체는 분홍색으로 표시하였다.

나머지 화면은 해당 객체의 매칭 score를 3가지 지표(I1~I3)로 보인 것이다.

3.3 매칭 점수에 대한 고찰

8

5_matching_every_shape.py



입력 영상: 'drawing2.png'

프로그램을 수차례 수행하여 0번, 2번, 1번 객체 순으로 거의 같은 모습으로 생긴 다른 객체와의 매칭 스코어를 관찰하여 보았다.

1번 객체: 최대 편차=2번 지표, 최소 편차=3번

2번 객체: 최대 편차=2번 지표, 최소 편차=3번

3번 객체: 최대 편차=2번 지표, 최소 편차=3번

고찰 사항: 그렇다면 2번 지표의 성능이 낮고, 3번 지표의 성능이 나쁘다고 할 수 있을까?

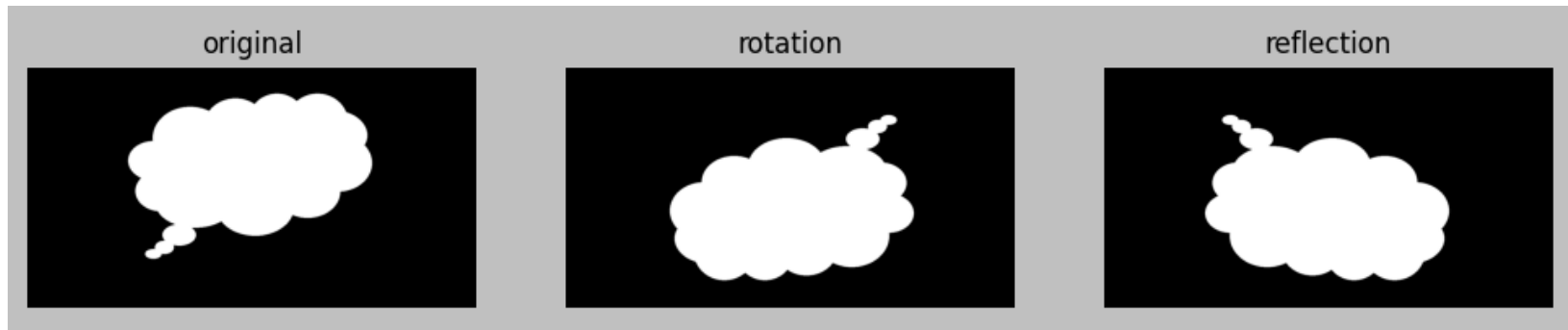
아니다!: 2번 지표는 전체적으로 지표 값이 크고, 3번 지표는 지표 값이 작은 경우를 생각해 보자.

➔ 그렇다면 **가장 합리적인 판단 기준**은 무엇일까? 이를 **프로그램으로 실증**해 보일 수 있을까. 같은 영상에 대해.. 혹은 **여러 영상**에 대해..

3.4 회전과 반사에 대한 휴 모멘트 변화 관찰

9

6__hu_moments_properties.py



```
Hu moments (original): '[ 1.92801772e-01  1.01173781e-02  5.70258405e-05  1.96536742e-06  
2.46949980e-12 -1.88337981e-07  2.06595472e-11]'
```

```
Hu moments (rotation): '[ 1.92801772e-01  1.01173781e-02  5.70258405e-05  1.96536742e-06  
2.46949980e-12 -1.88337981e-07  2.06595472e-11]'
```

```
Hu moments (reflection): '[ 1.92801772e-01  1.01173781e-02  5.70258405e-05  1.96536742e-06  
2.46949980e-12 -1.88337981e-07 -2.06595472e-11]'
```

- 원 영상과 회전 영상의 휴 모멘트 값이 일치하였다.
- 반사된 사례의 경우는 7번째 휴 모멘트의 값이 원 영상의 휴 모멘트와 부호만 반대고 크기는 같은 것이 확인된다.

4. 검토

10

- 휴 모멘트 계수를 이용한 매칭 기법에서 윤곽선을 이용한 매칭 기법과 이진 영상을 이용한 매칭 기법의 차이점(장단점)에 대해 기술하시오.
- 7개의 휴 모멘트를 연산하여 매칭스코어를 만들어 내는 방식과 이들 각각을 7차원의 벡터로 간주하여 머신 러닝의 학습기 혹은 딥 러닝의 학습기로 처리하여 매칭 여부를 판단하는 방식에 대해 구현 계획을 작성하여 보시오. ← 향후 학습의 내용이 필요.
 - 우선 검증해 볼 수 있는 실험 방안과 데이터 확보 방안에 기술하여 보시오.
 - 나머지 문제는 해당 섹션에서 다시 다루어 보겠습니다...