Goals

- To understand what are moments in image processing.
- To learn about Hu moments.
- To use the cv2.moments(), cv2.HuMoments() functions in python.

Theory

Working Principle

Introduction

Moments, in image processing, are weighted average of the image pixels' intensities. Weighted average is similar to normal average except some values from the set contribute more to the result than the others. The formula for spatial moments in a raster image is-

$$M_{i,j} = \sum_{x} \sum_{y} x^i . y^j . I(x, y)$$

Figure 1: Spatial Moments

Here, I(x,y) can have a value of either 0(background) or 1(foreground).

Using moments, one can obtain several properties of an image such as -

- **Area** :

$$M_{0,0} = \sum_{x} \sum_{y} x^0 . y^0 . I(x, y) = \sum_{x} \sum_{y} I(x, y)$$

Figure 2: Spatial Moments

- **Centroid** :

where

Central moments

$$(x_c, y_c) = \{M_{1,0} / M_{0,0} , M_{0,1} / M_{0,0} \}$$

Figure 3: Spatial Moments

$$M_{1,0} = \sum_x \sum_y x.I(x, y)$$

$$M_{0,1} = \sum_x \sum_y y.I(x, y)$$

Figure 4: Spatial Moments

Central moments are derived by reducing the spatial moments with centroid. They are computed in a center referenced frame, in the following manner-

However, a disadvantage of spatial and central moments is that they are dependant on the size of the object. So scaling is required. Area of the object is generally used as a scaling factor. So, central normalized moments are obtained by dividing the central moments with powers of area.

Hu Moments

Normalized moments are invariant to object size. However, they are affected by the orientation of the image. To overcome this, we can use Hu moments. Hu moments are invariant to translation, scaling and rotation.

Example

Consider the following image.

When we find moments using cv2.moments(), we get this result-

The Hu moments calculated will look like this-

$$\mu_{p,q} = \sum_x \sum_y (x - x_c).(y - y_c).I(x, y)$$

Figure 5: Spatial Moments

$$\eta_{p,q} = \mu_{p,q} \,/\, (M_{0,0})^{[(p+q)/2]\,+\,1}$$

Figure 6: Spatial Moments

$I_1 = \eta_{20} + \eta_{02}$
$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$
$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$
$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$
$I_5 = (\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2]+(3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2]$
$I_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$
$I_7 = (3\eta_{21}-\eta_{03})(\eta_{30}+\eta_{12})[(\eta_{30}+\eta_{12})^2-3(\eta_{21}+\eta_{03})^2]-(\eta_{30}-3\eta_{12})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2-(\eta_{21}+\eta_{03})^2].$

Figure 7: Spatial Moments

Figure 8: Spatial Moments

{'mu02': 8196768181.423641, 'mu03': 251425068011.5957, 'm11': 28364400148.0, 'nu
02': 0.08596224698563278, 'm12': 10288829414586.0, 'mu21': 171710226383.18262, '
mu20': 15356919975.078255, 'nu20': 0.16105315151257488, 'm30': 29489981567745.0,
 'nu21': 0.0032406157902109896, 'mu11': 14250377.54253006, 'mu12': -7712212987.1
41113, 'nu11': 0.0001494484712554994 6, 'nu12': -0.0001455493927765709, 'm02': 29
291682018.0, 'm03': 12192154557202.0, 'm00': 308793.0, 'm01': 80709118.0, 'mu30'
: -76361934603.83984, 'nu30': -0.0014411470781946316, 'nu03': 0.0047450408901973
71, 'm10': 108467643.0, 'm20': 53457620373.0, 'm21': 14153920964358.0}

Figure 9: Spatial Moments

[[2.47015398e-01]
 [5.63873328e-03]
 [2.57776208e-05]
 [6.62883183e-05]
 [-2.53517895e-09]
 [-4.60712574e-06]
 [1.03989371e-09]]

Figure 10: Spatial Moments

Applications

- Moments can be used to extract properties of an image.
- They are used in pattern recognition.
- Object identification also uses moments.
- Image analysis is done using moments.

Code

Moments

The main function we use to find moments is-

```
cv2.moments(array[, binaryImage])
```

where

`array : Raster image (single-channel, 8-bit or floating-point 2D array) or an array ( 1 \times N or  N \times 1 ) of 2D points (Point or Point2f ).`

`binaryImage : If it is true, all non-zero image pixels are treated as 1's. The parameter is used for images only.`

Hu Moments

The main function we use to find moments is-

```
cv2.HuMoments(m[, hu])
```

where

```
m : Input moments.

hu : Output Hu invariants.
```

Now, let us have a look at the code.

```
# Imports
import cv2

# Read image
img = cv2.imread("images/example/example.jpg")

# Convert to grayscale and apply thresholding
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
ret, th = cv2.threshold(gray, 127, 255, cv2.THRESH_BINARY)

# Find moments
M = cv2.moments(th, True) #Second parameter is true as th is a binary image

print M

# Find Hu invariant moments
Hu = cv2.HuMoments(M)

print Hu
```

Resources

- http://breckon.eu/toby/teaching/dip/opencv/SimpleImageAnalysisbyMoments.pdf
- https://en.wikipedia.org/wiki/Image_moment
- http://docs.opencv.org/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html
- http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/SHUTLER3/node8.html
- http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_imgproc/py_contours/py_contour_features/py_contour_features.html#contour-features

Exercises