1 Problem 1

Find the Fourier descriptor, $Z(k)$, of a circle of radius r.

Consider that the boundary of the circle is comprised of N evenly spaced points. The circle can be parametrized in the complex plane by the equation:

$$\begin{aligned} z(n) &= x(n) + i\,y(n) \\ &= r\cos\left(\frac{2n\pi}{N}\right) + i\,r\sin\left(\frac{2n\pi}{N}\right) \\ &= re^{2\pi i\frac{n}{N}} \end{aligned} \tag{1}$$

From this, the Fourier descriptor becomes

$$\begin{aligned} Z(k) &= \frac{1}{N}\sum_{n=0}^{N-1} re^{2\pi i\frac{n}{N}} e^{-2\pi i\frac{nk}{N}} \\ &= \frac{1}{N}\sum_{n=0}^{N-1} re^{2\pi i\frac{n}{N}(1-k)} && (2) \\ &= \frac{r}{N}\left(\frac{1-e^{2\pi i(1-k)}}{1-e^{\frac{2\pi i}{N}(1-k)}}\right) && (3) \end{aligned}$$

For $k = 1$, we can evaluate (2) to get

$$Z(1) = r.$$

However, if $k \neq 1$ then we need to use the expanded form in (3). Noticing that the denominator is nonzero for all such k, and evaluating the numerator, we get

$$Z(k) = 0 \qquad \forall k \neq 1.$$

Therefore the complete Fourier descriptor of a centered circle with radius r is

$$Z(k) = \begin{cases} r & k = 1 \\ 0 & k \neq 1. \end{cases}$$

2 Problem 2

Show the moments M_{10} and M_{01} are always 0.

Consider an image $I(x,y)$. Define

$$x_0 = \frac{\sum_{x,y} xI(x,y)}{\sum_{x,y} I(x,y)} \qquad y_0 = \frac{\sum_{x,y} yI(x,y)}{\sum_{x,y} I(x,y)}$$

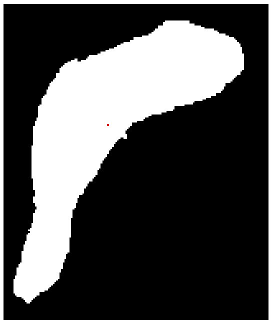
the moments are given by

$$\begin{aligned} M_{10} &= \sum_{x,y}(x-x_0)I(x,y) \\ &= \sum_{x,y} xI(x,y) - x_0\sum_{x,y} I(x,y) \\ &= \sum_{x,y} xI(x,y) - \sum_{x,y} xI(x,y) \\ &= 0 \end{aligned} \qquad \begin{aligned} M_{01} &= \sum_{x,y}(y-y_0)I(x,y) \\ &= \sum_{x,y} yI(x,y) - y_0\sum_{x,y} I(x,y) \\ &= \sum_{x,y} yI(x,y) - \sum_{x,y} yI(x,y) \\ &= 0 \end{aligned}$$

Thus for any image $I(x,y)$,

$$M_{10} = M_{01} = 0.$$

3 Problem 3



In this problem, I calculated the first 9 principal moments of the figure to the left. They are listed below in `moments.txt`. The code used to calculate these moments is split between `problem3.py` and `shapes.py`. Note that M_{02} is larger, as expected, because it coincides with the contribution from the spread of points in the y direction. The pixel highlighted in red is the center of mass of the image.

`moments.txt`

```
M00:    8657.0
M02:    15495144.7397
M20:    11372766.7294
```

`problem3.py`

```python
# Athanasios Athanassiadis Feb 2012
from shapes import *
from scipy.misc import imread,imsave

im = imread('img_moment.tif')
im *= 1.0/im.max()

M,CoM = get_moments(im, 3)
center = np.zeros(im.shape)
center[CoM[1],CoM[0]] = 1

imsave('img_moment.png', (im, im - center, im-center))
with open('moments.txt','w') as of_:
    of_.write('M00:\t{}\nM02:\t{}\nM20:\t{}'.format(M[0,0],M[0,2],M[2,0]))
```

4 Problem 4

Prove the invariance of $\frac{Z(k)}{Z(1)}$ for $k > 0$.

Consider an N-element contour $z(n)$, and a linear transformation of it $z'(n) = ce^{i\theta}z(n) + z_0$ which is scaled by a factor c, translated by a factor z_0, and rotated by an angle θ about its center. If the Fourier descriptor of z is $Z(k)$, then the Fourier descriptor of z' is given by

$$\begin{aligned} Z'(k) &= \frac{1}{N}\sum_{n=0}^{N-1} ce^{i\theta}z(n)e^{-2\pi i\frac{nk}{N}} + \frac{z_0}{N}\sum_{n=0}^{N-1} e^{-2\pi i\frac{nk}{N}} \\ &= ce^{i\theta}\sum_{n=0}^{N-1} z(n)e^{-2\pi i\frac{nk}{N}} \qquad \text{for } k > 0 \\ &= ce^{i\theta}Z(k) \end{aligned} \tag{4}$$

Thus, for $k > 0$,

$$\frac{Z'(k)}{Z'(1)} = \frac{Z(k)}{Z(1)}$$

which is independent of all transformation factors. Thus, $\frac{Z'(k)}{Z'(1)}$ is invariant to translation, rotation, and scaling.

5 Appendix: Common Code

The base code used for Problem 3 in this homework is contained in `shapes.py`.

`shapes.py`

```
# Athanasios Athanassiadis Feb 2012
import numpy as np

# calculate the (p,q)-moments of an image
# for p and q in range(n)
# x and y are reversed to account for row-col indexing
def get_moments(im, n):
    # initialize moments matrix
    M = np.zeros((n,n))

    # get list of pixels that are on and separate into list of x and y values
    pixlist = np.array(np.nonzero(im))
    y,x = pixlist

    # get the values at those points
    imvals = 1.0 * im[zip(pixlist)][0]
    N = len(imvals)
    if N==0:
        print 'get_moments: No figure found!'
        return []

    # CoM calculation
    y0,x0 = (pixlist * imvals).sum(1) / im.sum()

    # calculate all desired moments
    for p in range(n):
        for q in range(n):
            M[p,q] = ((x-x0)**p * (y-y0)**q * imvals).sum()

    return M,(x0,y0)
```