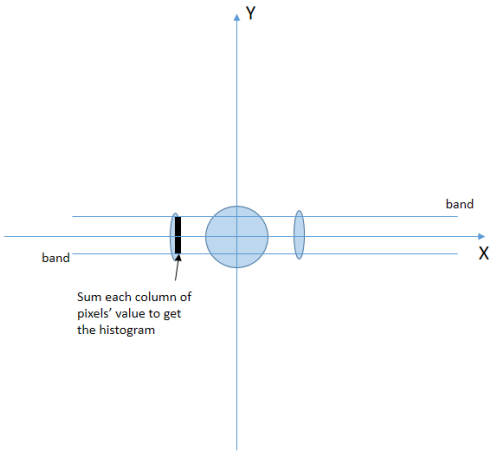
Bio-muscle project code document

1. Image process logic
   a) Find the center of the object
   b) From the center, get the eigenvectors of the object
   c) Find the larger eigenvector, and get the rotate angle by the larger eigenvector with x coordinate
   d) Rotate the object with the center and rotate angle
   e) Get the object width by the smaller eigenvalue
   f) Get the histogram in x coordinate by sum the pixels in each vertical direction
      Like:



   g) Use white top-hat to get rid of the background
      i. Build the kernel by default sigma, which can control by the user
   h) Find all peaks in the histogram after get rid of the background
      i. Smooth the histogram
      ii. Find all the apparent peaks
      iii. Find the closest peak to the center but distance to center is larger than the object width
      iv. Find the closest peak's symmetric peak
      v. If found, then regard these two peaks as the S10 peaks, else regards no effective peaks detected
      vi. From the S10 peaks, use the hexagonal pattern selection rule to compute the S11, S20 and so on
      vii. Use the computed S11, S20 and find the closest peak in the all apparent peaks
      viii. Get all the peaks' width in the histogram
      ix. Calculate σ_d by $\sigma_{h,k} = \sqrt{(\sigma_c^2 + \sigma_d^2 S_{hk} + \sigma_s^2 S_{hk}^2)}$, here $\sigma_{h,k}$ is the width of each peak, σ_c is set as 1.0 and σ_s is set as 0.0001
      x. Remove the slip part in the center
      xi. build the different number Gaussian Mixture Model(GMM) with the calculated

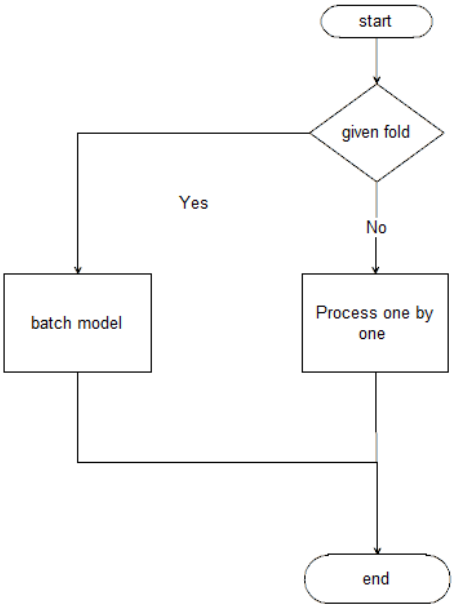
parameters

xii. calculate the parameters after the GMM fit

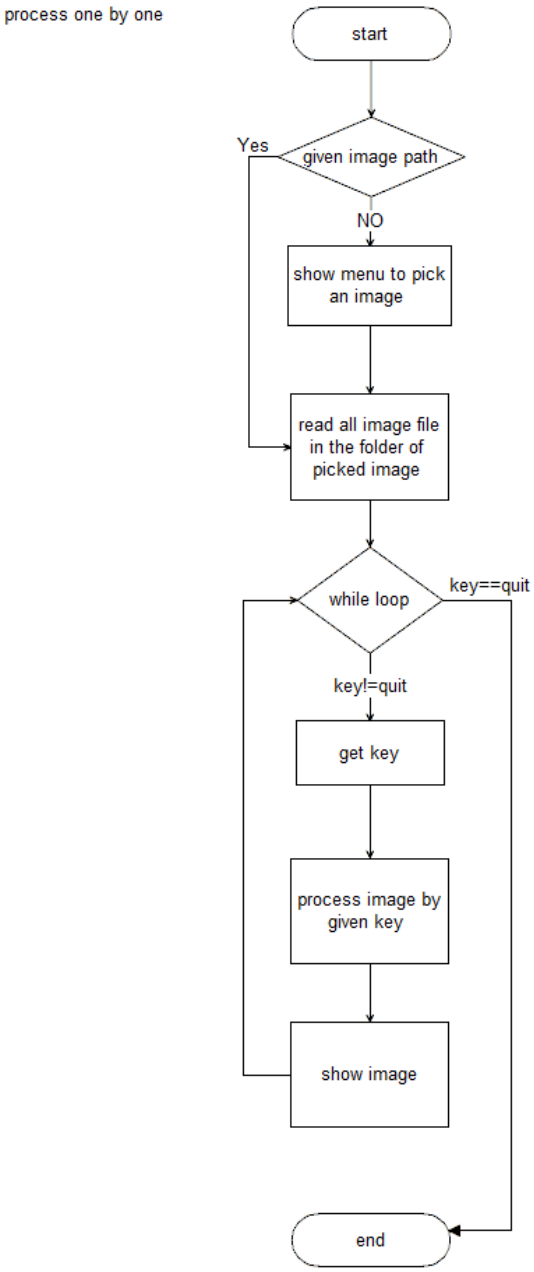
2. program logic

1) flow chart:

1) main:



2) Process one by one:



2) Key command:

.--show the next image

,--show the previous image

o--show the original image

c--show the center of object

e--show the larger eigenvector

m--increase the deltaMin and deltaMax

n--decrease the deltaMin and deltaMax

b--increase the deltaMin

v--decrease the deltaMin

B--increase the deltaMax

V--decrease the deltaMax

l--increase object band(effect after g is active)

k--decrease object band(effect after g is active)

t--get rid of the back ground(effect after g is active)

r--show the original rotate image

y--decrease the sigma

u--increase the sigma

g--show the histogram(effect after r is active)

i--save the histogram to a csv file(effect after t is active)

a--combine the function:r,g,t,p

p--detect the effective first pair of peaks(effect after t is active

s--show the smooth histogram

h--help

Esc--quit

3) bioImg class variable:
   1) imgName: image's file name
   2) imgCache: dictionary, store the image during the processing
   3) empty: whether the image is empty
   4) rotateThreshold: if the ratio of the two eigenvalues is larger than this variable, do not rotate the image
   5) color: histogram color
   6) symErr: error range during symmetric S10 peaks detection
   7) peakThres: only consider the peaks' height greater than this variable
   8) peakNum: number of peaks
   9) sigmaC: parameter used in Hexagonal pattern
   10) sigmaS: parameter used in Hexagonal pattern
   11) flags: a dictionary, control the program to calculate which parameters for the image; the flags variable is passed from one image to another when user switch to another image
      i. center: center of the project
      ii. ev: eigenvector
      iii. hist: histogram
      iv. rotate: rotate the image

v. org: show the original image
vi. bg: get rid of the background
vii. peak: detect the S10 peak
viii. smoothHist: smooth the bg subtract histogram
ix. deltaMin: use for convert image
x. deltaMax: use for convert image
xi. sigma: top white hat Gaussian Kernel's size
xii. 2ndsigma: second top white hat(not used any more)

12) Info: store the calculated parameters during the processing, see detail in the "BioImg info structure.txt"

3. Global Function

1) inverte(imagem):

Parameter:

Imagem: image

Function: invert the uint8 image

Return: inverted image

2) thresholdImg(img,percent):

Parameter:

Img: image

Percent: percent of the valid pixels

Function: read the image's histogram, and for the given percent*all the pixels, find the threshold value of the histogram

Return: threshold value

3) noBoundImg(img):

Parameter:

Img: image

Function: discard the edge of the image, because we assume that the valid object is at the center of the image

Return: image

4) fullPath(filePath,fileName):

Parameter:

filePath: file path to folder

fileName: file name

Function: combine the file path with the file name to get the full file path

Return: full file path

5) GM(x,u,sigmad,alpha):

Parameter:

x: array

u: μ, the center of the Gaussian Model

sigmad: computed sigmad, the standard deviation of the Gaussian function

alpha: $\sqrt{2\pi}$ *H*var (H is the height of the peak, var is the peak's width

Function: single Gaussian Model

Return: x array with computed GM

6) BGND(x,u,sigmad,alpha,thresh):

Parameter:

x: array

u: μ, the center of the Gaussian Model

sigmad: computed sigmad, the standard deviation of the Gaussian function

alpha: $\sqrt{2\pi}$ *H*var (H is the height of the peak, var is the peak's width

thresh: the distance from the center which in this distance are junk part of the histogram

Function: second background subtract with Gaussian function which discard the center junk part

Return: x array

7) GMM2(x,S10,sigmad10,centerX,alpha10,S11,sigmad11,alpha11,sigmaX,alphaX,thresh):

Parameter:

x: array

S10: S10 value

Sigma10: peak S10's width

centerX: center

alpha10:S10's alpha value

S11: S11 value

Sigma11: peak S11's width

alpha11:S11's alpha value

sigmaX: subtract background Gaussian function's standard deviate

alphaX: subtract background's alpha value

thresh: the distance from the center which in this distance are junk part of the histogram

Function: Gaussian Mixture Model with 2 peaks at each side(left and right) and also do the second time subtract background

Return: x array

8) GMM2nobgnd(x,S10,sigmad10,centerX,alpha10,S11,sigmad11,alpha11,sigmaX,alphaX,thresh):

Parameter:

x: array

S10: S10 value

Sigma10: peak S10's width

centerX: center

alpha10:S10's alpha value

S11: S11 value

Sigma11: peak S11's width

alpha11:S11's alpha value

sigmaX: subtract background Gaussian function's standard deviate

alphaX: subtract background's alpha value

thresh: the distance from the center which in this distance are junk part of the histogram

Function: Gaussian Mixture Model with 2 peaks at each side (left and right) and without do the second time subtract background

Return: x array

9) GMM3(x,S10,sigmad10,centerX,alpha10,S11,sigmad11,alpha11,S20,sigmad20,alpha20, sigmaX,alphaX,thresh):

Parameter:

x: array
S10: S10 value
Sigma10: peak S10's width
centerX: center
alpha10:S10's alpha value
S11: S11 value
Sigma11: peak S11's width
alpha11:S11's alpha value
S20: S20 value
Sigma20: peak S20's width
Alpha20:S20's alpha value
sigmaX: subtract background Gaussian function's standard deviate
alphaX: subtract background's alpha value
thresh: the distance from the center which in this distance are junk part of the histogram

Function: Gaussian Mixture Model with 3 peaks at each side (left and right) and also do the second time subtract background

Return: x array

10) GMM4(x,S10,sigmad10,centerX,alpha10,S11,sigmad11,alpha11,S20,sigmad20,alpha20, S21,sigmad21,alpha21,sigmaX,alphaX,thresh):

Parameter:

x: array
S10: S10 value
Sigma10: peak S10's width
centerX: center
alpha10:S10's alpha value
S11: S11 value
Sigma11: peak S11's width
alpha11:S11's alpha value
S20: S20 value
Sigma20: peak S20's width
Alpha20:S20's alpha value
S21: S21 value
Sigma21: peak S21's width
Alpha21:S21's alpha value
sigmaX: subtract background Gaussian function's standard deviate
alphaX: subtract background's alpha value
thresh: the distance from the center which in this distance are junk part of the

histogram

Function: Gaussian Mixture Model with 4 peaks at each side (left and right) and also do the second time subtract background

Return: x array

11) isImg(fileName):

Parameter:

fileName: file name

Function: determine whether the file is an image

Return: true or false

12) isRotatedImg(fileName):

Parameter:

fileName: file name

Function: determine whether the file is a rotated image by the program

Return: true or false

13) writeFile(bioImg,filePath,imgList,fNum):

Parameter:

bioImg: current image's objective

filePath: file path

imgList: all the image name list in the folder

fNum: image number in the imgList

Function: write the information and rotated image to the same folder

Return: None

14) printHelp():

Parameter:

None

Function: print the help

Return: None

15) openFile(pickFile=None) :

Parameter:

pickFile: default is None, the picked image's file path

Function: main process function, including the while loop to handle the image. In the while loop, get the key value and set the flags, then call the function to process the image.

Return: None

16) bioQuit():

Parameter:

None

Function: quit the program

Return: None

4. bioImg class function

1) bkImg(self,img):

Parameter:

Img: image

Function: get the threshold pixel value of the center part image only consider the pixels with value larger than the threshold and get rid of the rod of the object by using

morphologyEX() funtion

Return: image

2) centerImg (self):

Parameter:

None

Function:

Find all non-zero pixels, if there are less than 100 pixels larger than 0, then regard the image as empty.

Call the bkImg(), then use the moment to get the center location.

Store the center location into the self.info.

Return: None

3) evMImg (self):

Parameter:

None

Function:

Call the bkImg().

Get all the non-zero pixels.

Get each non-zero pixels' relative position to the center.

Get the correlation matrix.

Use np.linalg.eig() to get the eigenvectors and the eigenvalue.

Get the rotation angle with the eigenvectors.

Get the rotation matrix M with the rotation angle and the center location.

Return: None

4) bandHist(self,img):

Parameter:

Img: image

Function: sum the value of each pixel in x direction, then normalize the value in order to show the histogram on the right side of the image.

Return: None

5) hist(self,img,noBG=False,lenX=0):

Parameter:

Img: image

noBG: default is false, in order to determinate the image with background or not

lenX: default is 0, no longer use

Function: sum the value of each pixel in y direction to get the histogram in x coordinate, and normalize it. For given different noBG flags, store the histogram to different key.

Return: None

6) drawHist(self,img,histVal):

Parameter:

Img: image

histVal: histogram

Function: for a given histogram, draw it on the image. Also draw the band line of the object

Return: None

7) kernelXY(self,sigma=0):

Parameter:

Sigma: default is 0, control whether to use the default sigma in the class or not

Function: build a 2D Gaussian kernel

Return: 2D array

8) whiteTopHat(self,sigma=5,first=True):

Parameter:

Sigma: default is 5, build the tmpkernel for the filter2D() function

First: default is True, no longer use

Function: build the kernel and then use the white_tophat() with kernel to do the background subtraction

Return: None

9) getRightPeaks(self):

Parameter:

None

Function: main function of peak detection and model fit.

1. Find all the apparent peaks, store into the peak_list
2. Then in the peak_list, find the closest peak which is also larger than the object width
3. Find the symmetric peak
4. Use the hexagonal pattern selection rule, get all the valid peaks
5. Get the width of these peaks
6. Base on the S_{hk} and the σ_{hk}, calculate the σ_d
7. Remove the slip in the center junk part, in order to show the histogram without the junk part
8. Use GMM to fit the histogram
9. Calculate the necessary parameters from the GMM fitting model

Return: None

10) findAllPeaks(self):

Parameter:

None

Function: find all apparent peaks:

1. smooth the histogram
2. get the derivative of the histogram
3. find the signal changed point, regard as the candidate peaks
4. merge the neighbor candidate peaks, and regard the middle as the peak location

Return: None

11) findMostSymatric(self,p1,p_list,center):

Parameter:

P1: already found peak

P_list: all the candidate peaks list

Center: center of the object

Function: according to the p1 and the center, find the most symmetric peak on the other

side.

Return: if found, return the peak location, else return None

12) calculateParameterOfEachPeak (self):

Parameter:

None

Function: calculate the parameters (mean, standard derivative, area)of each peak after the GMM model fit, using the result of the fitting

Return: None

13) fitGMM (self):

Parameter:

None

Function: fit the histogram with GMM, depends on the different peaks user defined. Draw the fitting histogram and original histogram with matplotlib.pyplot package.

Return: None

14) calculateSigmaD (self):

Parameter:

None

Function:calculate the σ_d of each peak, and get the mean of them.

Return: None

15) widthDetect (self,peaks,hist,widthList):

Parameter:

Peaks:all the detected peaks location

Hist: histogram

widthList: store the width of each peak

Function:from the histogram and the given peak location, find the peak's width(standard derivative)

Return: None

16) widthOfPeaks (self):

Parameter:

None

Function:call the widthDetect() function to compute each peak's width(standard derivative).

Return: None

17) findClosestPeak (self,s,peakList):

Parameter:

S: from the hexagonal pattern selection rule calculated theory location

peakList: all the apparent candidate peaks

Function: from the theory location, find the peak location in the peakList within a given range.

Return: None or found peak

18) hexagonalPattern(self,symP,peak,peakList):

Parameter:

symP: symmetric peak location

peak: peak location

peakList: all the apparent candidate peaks

Function: use the hexagonal pattern selection rule, compute the S10, S11 … values. And then in the candidate peak list find the closest peaks of these computed values.

Return: None

19) removeSlip (self,hist):

Parameter:

Hist: the histogram

Function: get the rid of the junk slip in the center part of the histogram

Return: None

20) cmdcal (self):

Parameter:

None

Function: for the set flags, calculate the relative parameters.

Return: None

21) cmdshow (self):

Parameter:

None

Function: for the set flags, show the image

Return: None

22) convertImg (self,img):

Parameter:

Img: image

Function: scaling the image, for user to see the object obviously (for represent only)

Return: pix8 image

23) normalizeHist (self,hist,,xCoor=True,lenX=0):

Parameter:

hist: histogram

xCoor: default if True, define whether the histogram is shown on X coordinate or Y coordinate

lenX: default is 0, no longer use

Function: normalize the histogram for showing on the image (for represent only)

Return: histogram array