Image processing via model matching using Python

Sathishkumar Matheswaran*, Md. Abdullah al Kamran Ripon**

Hochschule Furtwangen University
*sathishkumar.matheswaran@hs-furtwangen.de
**md.abullah-al-kamran.ripon@ hs-furtwangen.de

Abstract: This work demonstrates a method to find out good image information, pixel values, of ultrasound image. Ultrasound images scanned to find out best pixel values available within an image area. Image area is defined automatically by selecting nonzero pixel values on the image. After that, program will scan through the image area for best available pixel values. With these Co-ordinate values, angle & radius of a pie shape mask will be calculated and then image will be cropped with this mask.

Keywords*:* Model matching, ultrasound imaging, pixel values, python.

1. INTRODUCTION

This paper describes image processing of ultrasound images via model matching. Main goal of this work is to develop a quality measure to identify how well images matches are, also a heuristic is developed which can find out best image information or pixel values of an image.

2. MATERIALS AND METHODS

In this work Python 2.7.1 software used to develop the source code and several Python image modules, including numpy, Matplotlib, os, sys, PIL and math are used to develop source code.

At first necessary image modules are imported from python library. Then using below code, directory is defined to load images for further process. This method can process multiple images at the same time. 25 images were processed in this method.

```
for infile in os.listdir('.'):
    if infile.endswith('.png'):
        matrix = img.imread(infile)
        m = Image.open(infile)
        fn, ftext = os.path.splitext(infile)
```

After importing necessary images from the directory, next step is to define the image area. To define a correct image area it is important to find out right coordinate points. For ultrasound image three coordinate points requires to define the image area. To find out right coordinate points source code will look for first nonzero points on the image (top, left and right side). Original ultrasound image already have many unnecessary nonzero points which might leads to choose wrong coordinate points. To find a correct coordinate point a range within the image is defined. Here `i` and `j` represents X and Y axis.

```
a = []
        for j in range(0, m2 / 3):
            for i in range(m1 / 3, 3 * m1 / 4):
                if sum(m.getpixel((i, j))) >= 255:
                    a.append((i, j))
                    break
        a = a[0]
        [C1, C4] = a

        b = []
        for i in range(0, m1 / 4):
            for j in range(m2 / 2, 4 * m2 / 5):
                if sum(m.getpixel((i, j))) >= 30:
                    b.append((i, j))
                    break
        b = b[0]
        [C2, C5] = b

    c = []
        for i in range(12 * m1 / 13, m1/3, -1):
            for j in range(12*m2 / 13, m2 / 3, -1):
                if sum(m.getpixel((i, j))) >= 30:
                    c.append((i, j))
                    break
        c = c[0]
        [C3, C6] = c
        print "Detected Co-ordinates values
:","centre',(C1,C4),'Left', (C2,C5),'Right',
(C3,C6)
```

After collecting necessary coordinate values, next step is to scan through the image to find available best pixel values. Below code will scanned through the image area for best pixel values and makes a list.

```
a = 0
for i in range (-4, 5):
    for j in range (-4, 5):
        Cx = C1 + i/4
        Cx1 = C2 + i
        Cx2 = C3 + i
        Cy = C4 + j
        Cy1 = C5 + j
        Cy2 = C6 + j
```

After that using these coordinate values, program will start calculating radius and angel of the pie shape mask. Below code calculate angle and radius of a pie shape mask.

```
def Angle_Radius(Cx, Cy, Cx1, Cy1, Cx2, Cy2):
#Setting Desired co-ordinates as origin
    Cx1_mod = Cx - Cx1
    Cy1_mod = Cy - Cy1
    Cx2_mod = Cx - Cx2
    Cy2_mod = Cy - Cy2

   #   ANGLE Calculation

    Angle1 = math.atan2(Cy1_mod - 0, Cx1_mod - 0)
    Angle2 = math.atan2(Cy2_mod - 0, Cx2_mod - 0)
    Angle = abs(Angle2) - abs(Angle1)
    Angle11 = math.degrees(Angle1)
    Angle22 = math.degrees(Angle2)

#   Radius Calculation
    Radius1 = ((Cx - Cx1) ** 2 + (Cy - Cy1) ** 2)
    Radius1 = math.sqrt(Radius1)
    Radius2 = ((Cx - Cx2) ** 2 + (Cy - Cy2) ** 2)
    Radius2 = math.sqrt(Radius2)
```

```
    if Radius1 > Radius2:
        Radius = math.floor(Radius1)
    else:
        Radius = math.floor(Radius2)
    return Radius, Angle11, Angle22
```

After calculating angle and radius of mask, next step is to create a mask based on radius and angle calculated in the previous step. Below code will import calculated angle and radius values.

```
from Task22 import Angle_Radius
Radius, Angle11, Angle22 = Angle_Radius(Cx,
Cy, Cx1, Cy1, Cx2, Cy2)
```

Below code will create pie shape mask.

```
def Pie_mask(shape, centre, Radius, Angle):
    x, y = np.ogrid[:shape[0], :shape[1]]
    cx, cy = centre
    Anglemin, Anglemax = np.deg2rad(Angle)

    if Anglemax < Anglemin:
        Anglemax += 2 * np.pi

    r2 = (x - cx)*(x - cx)+(y - cy)*(y - cy)
    theta = np.arctan2(x-cx,y-cy) - Anglemin
    theta %= (2 * np.pi)
    cirmask = r2 <= Radius * Radius
    anglemask = theta<=(Anglemax - Anglemin)
    return cirmask * anglemask
```

After creating the mask, pixels of the image that covers under the mask will be accessed and summed to get the total pixel value of that image. Using different mask size and shape, best pixel values will be calculated and the corresponding mask will be used for cropping the image.

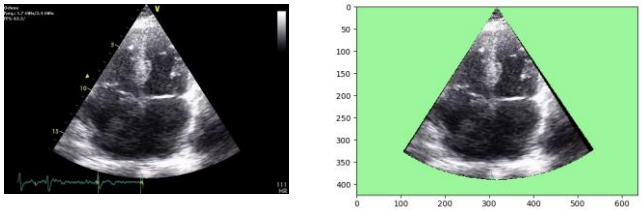
In the next step, pie mask is imported and crop the image where it has best image information. Below code import pie-mask and do the necessary cropping.

```
from Task33 import Pie_mask
centre = (Cx, Cy)
Angle = (Angle11, Angle22)
mask = Pie_mask(matrix.shape, (Cy, Cx),
Radius, (-Angle11, -Angle22))
a = matrix[mask]
a1 = np.sum(a)
print "a1 values are", a1
if a1 > a:
    a = a1
    best_Cx = Cx
    best_Cy = Cy
    best_Cx1 = Cx1
    best_Cx2 = Cx2
    best_Cy1 = Cy1
    best_Cy2 = Cy2
```

After cropping the image, it will save in a certain directory.

3. RESULTS

Below show original and resulted image after cropping.



a b

Fig. 2. a. Original image. b. Resulted image.

4. DISCUSSION

In this work, main goal was to develop a method to find out best image information of ultrasound images based on model matching. To do that image area is defined by finding nonzero pixels on ultrasound images by defining ranges within the image. Position of image has some effects on outcomes. If image centre (origin) not in the image i.e. out of the border, it leads to poor cropping. Images without proper edges results in reduced angle of mask. Images with texts and figures also results improper masking.

5. FUTURE WORK

- An edge detection method can be introduced for selecting better image area.
- Creation of an algorithm to avoid the texts, colour bar and any graphs.

7. REFERENCES

1. http://effbot.org/imagingbook/pil-index.htm

2. https://docs.python.org/3/tutorial/