Practical 4 Part 2 Classification, matching, regionprops()

March 2, 2021

Jaya Parmar, MATLAB filenames FLIRsupervised.m, geomshapes.m

1 Introduction

We have two aims for this report. Our first aim is to classify a FLIR image using supervised classification so that the hot object is seperated from the rest of the image objects.
Our second aim is to find and count circular objects from different geomatric shapes using regionprops().
We will divide the report into two sections to show these applications

2 Method

2.1 FLIR image

We start by reading in the infrared images FLIR0163.jpg and RGB image FLIR0163-photo.jpg.

$$IR = imread('FLIR0163.jpg') \tag{1}$$

$$RGB = imread('FLIR0163 - photo.jpg') \tag{2}$$

See the IR and RGB images in Figure 1 and Figure 2 in Results section.

$$figure(1), imshow(IR), title('IR\ image'), impixelinfo \tag{3}$$

$$figure(2), imshow(RGB), title('RGB\ image'), impixelinfo \tag{4}$$

Seperating out the red band from the color images.

$$i = RGB(:, :, 1) \tag{5}$$

$$j = IR(:, :, 1); \tag{6}$$

and checking their histograms shown in Figure 3.

$$figure(3), subplot(2,1,1), imhist(j), title('IR\ histogram(redband)') \quad (7)$$

$$, subplot(2,1,2), imhist(i), title('RGB\ histogram(redband)') \quad (8)$$

We can see that it is difficult to find a segment in the histograms to seperate into 4 groups.

We would use alternate method for classification as follows.

Create a matrix 'V' with columns as the red bands of RGB and IR image.

$$, V = double([i(:), j(:)]) \quad (9)$$

We are required to seperate the image into 4 groups and each group has 5 pixel pairs. Hence we need a sample of 20 pixel pairs.

We begin by looking at the RGB image in Figure 1 and manually extract the coordinates of the hot object using color information. We make four color groups starting from outermost yellow to innermost whitish pink.

Our four groups will be yellow, dark red, pink and whitish pink.

From these four groups, we pick 5 pixel pairs each as

- Yellow - Group 1
  Pixel Pair (120, 48) - Pixel value from IR image is 205 (see pixel value from Figure 1).
  In the Matlab command window, we write RGB(120,48) to get pixel value from RGB image as 2.
  We can type ir(48,120) in Matlab command window, which will also give pixel value 205.

  This way we find four more pixel pairs and note their pixel values in IR and RGB image.
  Pixel pair Pixel value(RGB,IR)
  120,48 2 205
  200,47 4 211
  113,204 31 214
  197,209 25 202
  160,212 14 227

- Dark Red - Group 2
  Pixel pair Pixel value(RGB,IR)
  131,65 2 236
  190,54 10 233

189,185 27 231
122,178 29 233
159,54 2 240

- Pink - Group 3
  Pixel pair Pixel value(RGB,IR)
  162,79 4 226
  134,156 30 230
  179,160 27 247
  193,123 5 227
  129,113 2 245

- Whitish pink - Group 4
  Pixel pair Pixel value(RGB,IR)
  161,100 4 234
  159,143 29 240
  145,121 4 214
  176,120 6 243
  161,136 27 212

Storing the groups in column vector g and the above pixel values in 20X2 matrix t.

$$g = [1\ 1\ 1\ 1\ 1\ 2\ 2\ 2\ 2\ 2\ 3\ 3\ 3\ 3\ 3\ 4\ 4\ 4\ 4\ 4]' \tag{10}$$

$$t = [2\ 2054\ 21131\ 21425\ 20214\ 2272\ 23610\ 23327\ 23129\ 2332\ 2404\ 22630\ 23027\ 2475\ 2272\ 2454\ 23429\ 2404\ 2146 \tag{11}$$

Now is the time to classify the red band of RGB and IR images into the 4 groups based on the pixel values provided. Matlab has 'classify' function to do this.

$$class = classify(V, t, g) \tag{12}$$

Convert the vector V to a matrix using function 'vec2mat'.

$$a = vec2mat(class, size(i, 1)) \tag{13}$$

$$b = ind2rgb(a', prism) \tag{14}$$

Conver the 320X240 double indexed image 'a' into a RGB image using 'ind2rgb' function.

$$b = ind2rgb(a', prism) \tag{15}$$

and show the classified image in Figure 4.

$$figure(4), imshow(b), title('Supervised'), impixelinfo \tag{16}$$

The last part of this puzzle is to seperate out the hot object from this classified image.
Function regionprops will help us achieve this.
To use this function, the classified image needs to be converted to a binary image first.

$$BW = im2bw(b) \tag{17}$$

BW is the binary image which can be fed to nwlabel to extract the connected components of the image

$$L = bwlabel(BW, 8); \tag{18}$$

We will use regionprops now with the basic option since we are interested only in the area property of these regions.

$$regionMeasurements = regionprops(L,'basic') \tag{19}$$

$$allAreas = [regionMeasurements.Area]; \tag{20}$$

allAreas is a row vector containing the areas of the 24 regions in the image. We want the biggest area from these 24 areas.Hence we sort allAreas in descending order and take the biggest area as below.

$$[sortedAreas, sortIndexes] = sort(allAreas,'descend') \tag{21}$$

$$biggestArea1 = sortedAreas(1:1) \tag{22}$$

We make a new matrix newBiggest that contains the coordinates of the biggestArea1.

$$newBiggest = (L == sortIndexes(1)) \tag{23}$$

Converting this matrix to double and multiplying with the binary image BW

$$double(newBiggest) \tag{24}$$

$$imBiggest = double(BW). * newBiggest \tag{25}$$

The new image imBiggest is shown in Figure 5 in results section.

Figure 5 in the Results section shows the resulting regions BW1, BW2 and BW3.

$$figure(5), imshow(imBiggest, []), title('Object\ in\ question'), impixelinfo \tag{26}$$

2.2 Geomatrical shapes

Read in the images below and convert it to gray image

$$im = imread('geom_shapes.jpg') \tag{27}$$

$$figure(1), imshow(bw), title('Search\ image'), impixelinfo \tag{28}$$

$$im = rgb2gray(im) \tag{29}$$

We find the entropy of this grayscale image

$$E = entropyfilt(im) \tag{30}$$

and rescale the image

$$Eim = rescale(E) \tag{31}$$

We need to binarize the image and check the background and foreground as below

$$bwim = imbinarize(Eim) \tag{32}$$

Everything that is not bwim is background

$$bg = (\ bwim) \tag{33}$$

Show the foreground and background in Figure 1 as below

$$figure(1), subplot(2,1,1), imshow(bwim), title('Foreground') \tag{34}$$

$$subplot(2,1,2), imshow(bg), title('Background'), impixelinfo() \tag{35}$$

Note that the pixel values are inverted. We wish foreground white and background black.
We will use 'bg' instead of 'bwim' for the bwlabel function.

$$[labeled, num] = bwlabel(bg, 8) \tag{36}$$

The labeled image is fed to regionprops to get the properties of the regions. This time we use 'all' properites because we want the circularity property which is not included in the basic properties.

$$regions = regionprops(labeled,'all') \tag{37}$$

$$regionSize = size(regions) \tag{38}$$

Extract the perimeter and area properties for all regions since the circularity formula is dependent on these properties.

$$allPerimeters = [regions.Perimeter] \tag{39}$$

$$allAreas = [regions.Area] \tag{40}$$

$$allCircularities = allPerimeters.^2./(4 * pi * allAreas) \tag{41}$$

Sort the circularities in descending order because the objects with circularity values near to 1 are circles i.e. our objects of interest.

$$[sortedCircularities, circleIndexes] = sort(allCircularities,' descend') \tag{42}$$

Now we run a loop to scan through all 30 regions in the image. The properties of regions with allCircularities less than 1.19 are stored in a row vector 'matches'. The counter 'matchIndex' will count the circular regions to be stored in 'matches'. 'matchIndex' is initialized to 1

$$matchIndex = 1; \tag{43}$$

$$fori = 1 : num \tag{44}$$

$$ifallCircularities(i) < 1.19 \tag{45}$$

$$matches(matchIndex) = regions(i) \tag{46}$$

$$matchIndex = matchIndex + 1 \tag{47}$$

Now within the loop we plot the original grayscale image im and make bounding boxes on im for the regions that satisfy the condition allCircularities(i) ¡ 1.19.
We choose 1.19 value by trial and error method to obtain best possible results.

$$figure(2), imshow(im) \tag{48}$$

hold on

$$forj = 1 : length(matches) \tag{49}$$

$$rectangle('Position', matches(j).BoundingBox,' EdgeColor',' r') \tag{50}$$

end end end

Now we have found the circular objects shown in Figure 2

To count the marked objects from Figure 2, we will use counter 'matchIndex' from above. Since it starts at value 1, our number of circular objects is one less than 'matchIndex'.

$$circularobjects = matchIndex - 1 \tag{51}$$

$$disp(circularobjects) \tag{52}$$

3 Results

3.1 FLIR image

Figure 1

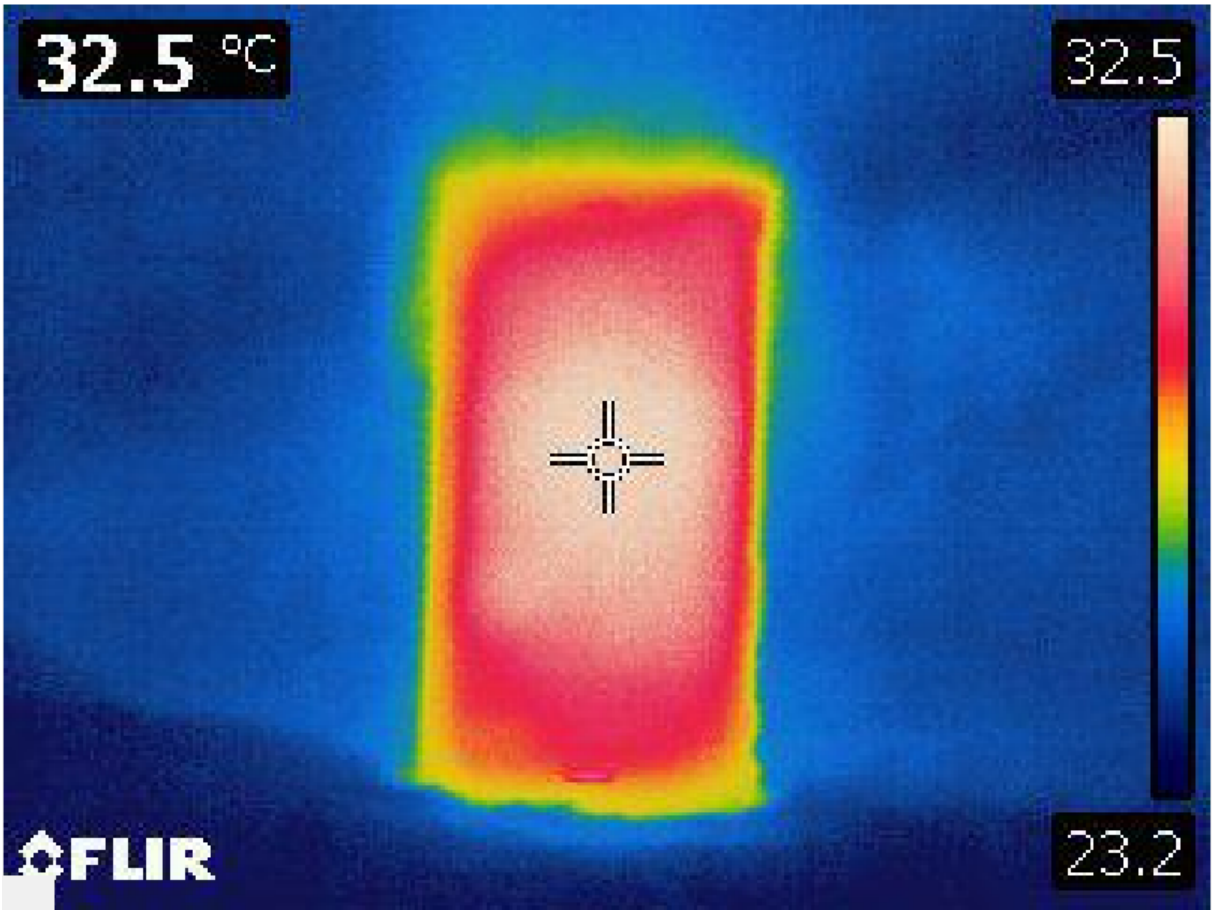


Figure 2

Figure 3

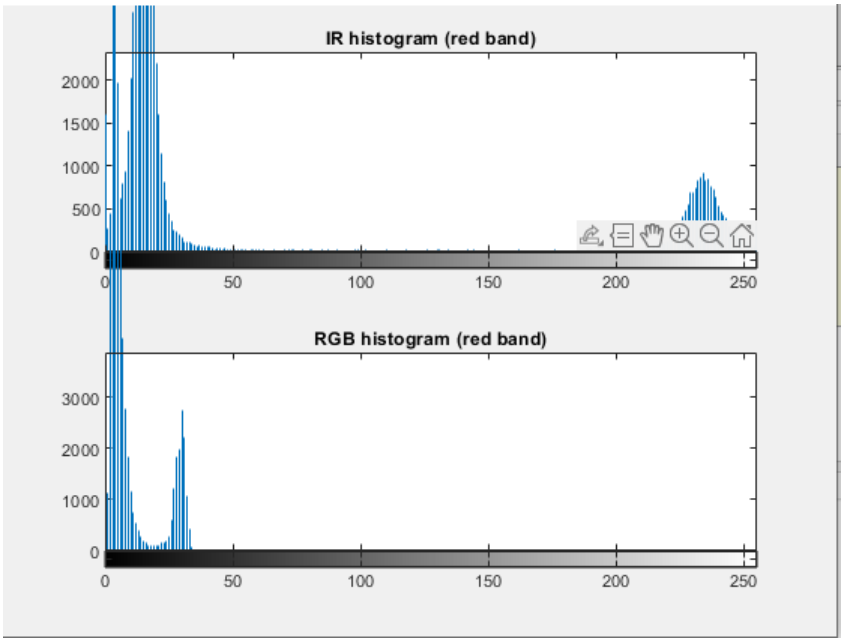

IR histogram (red band)
2000
1500
1000
500
0
0
50
100
150
200
250
RGB histogram (red band)
3000
2000
1000
0
0
50
100
150
200
250

Figure 4

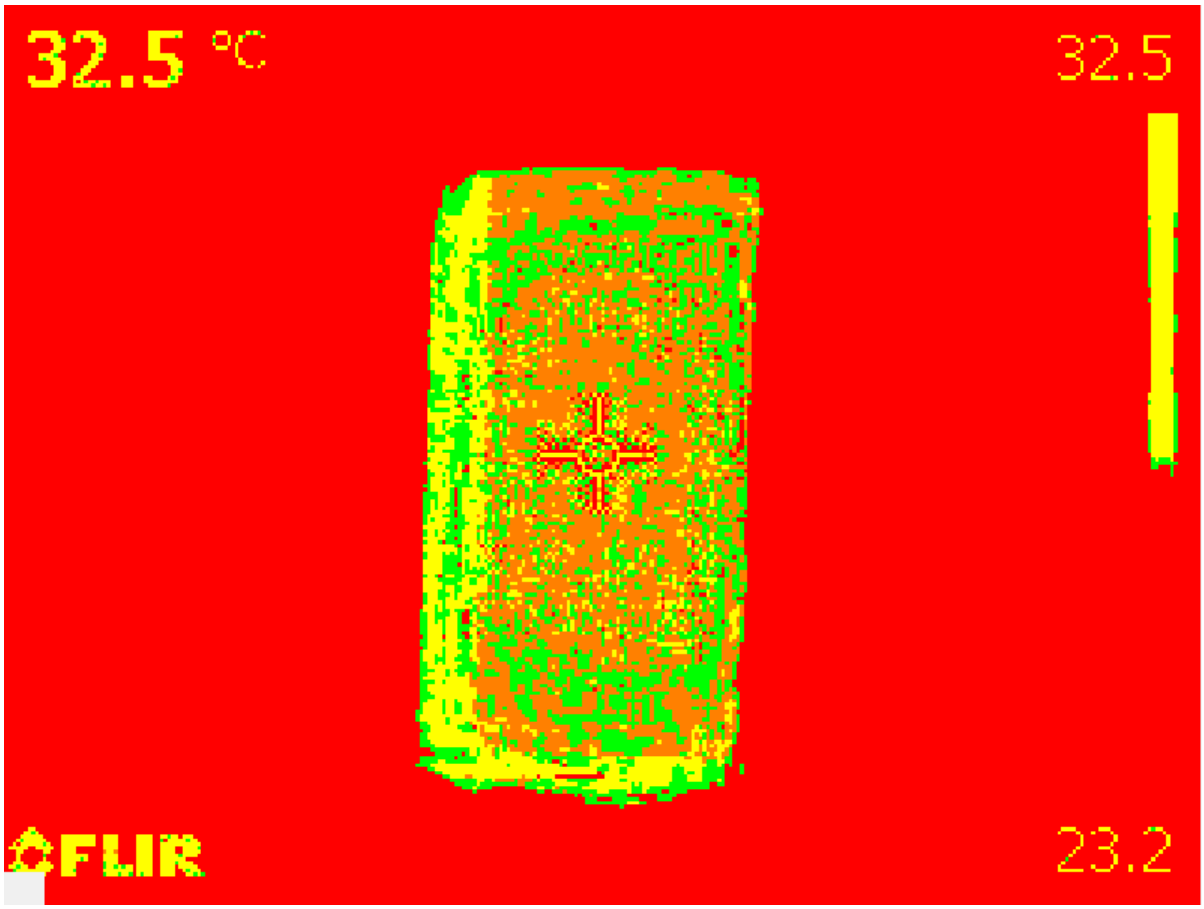
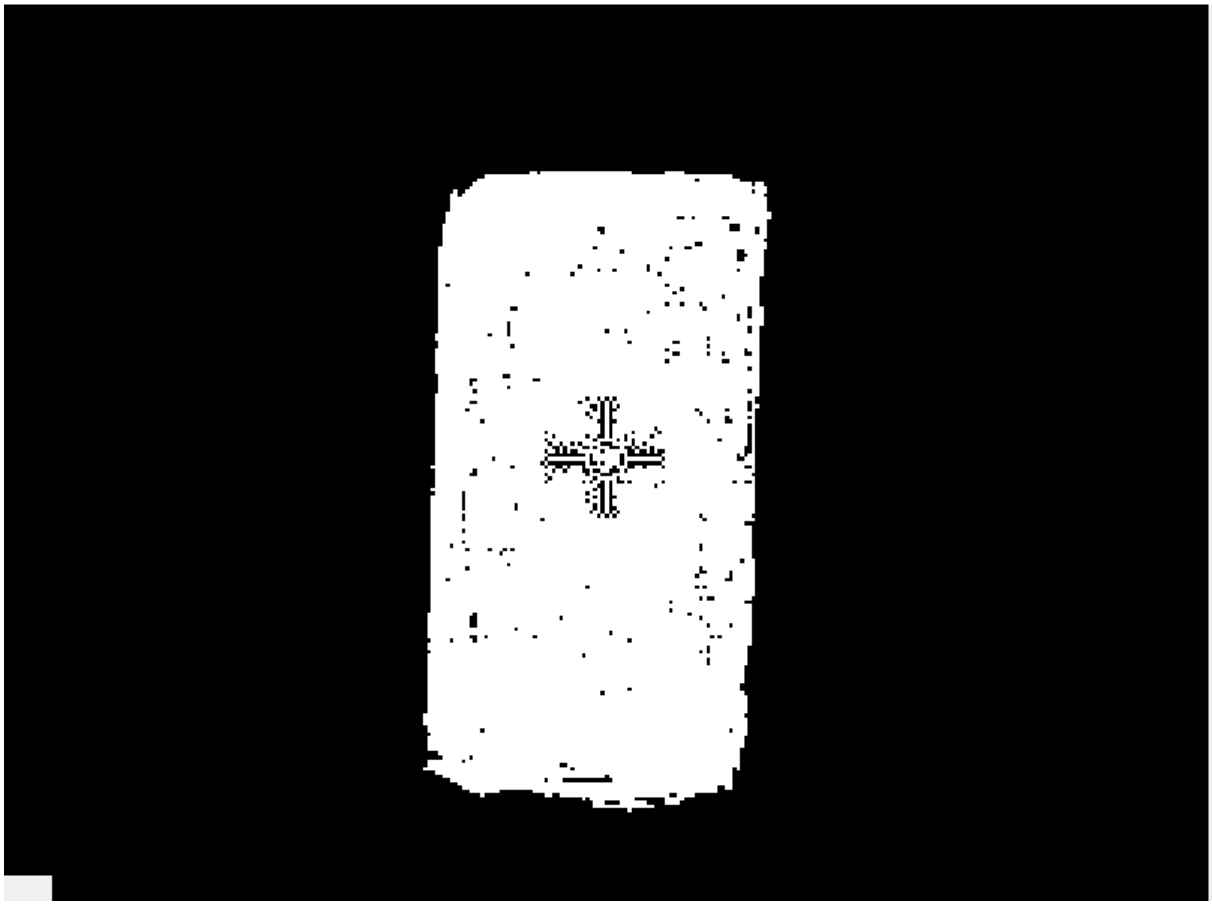


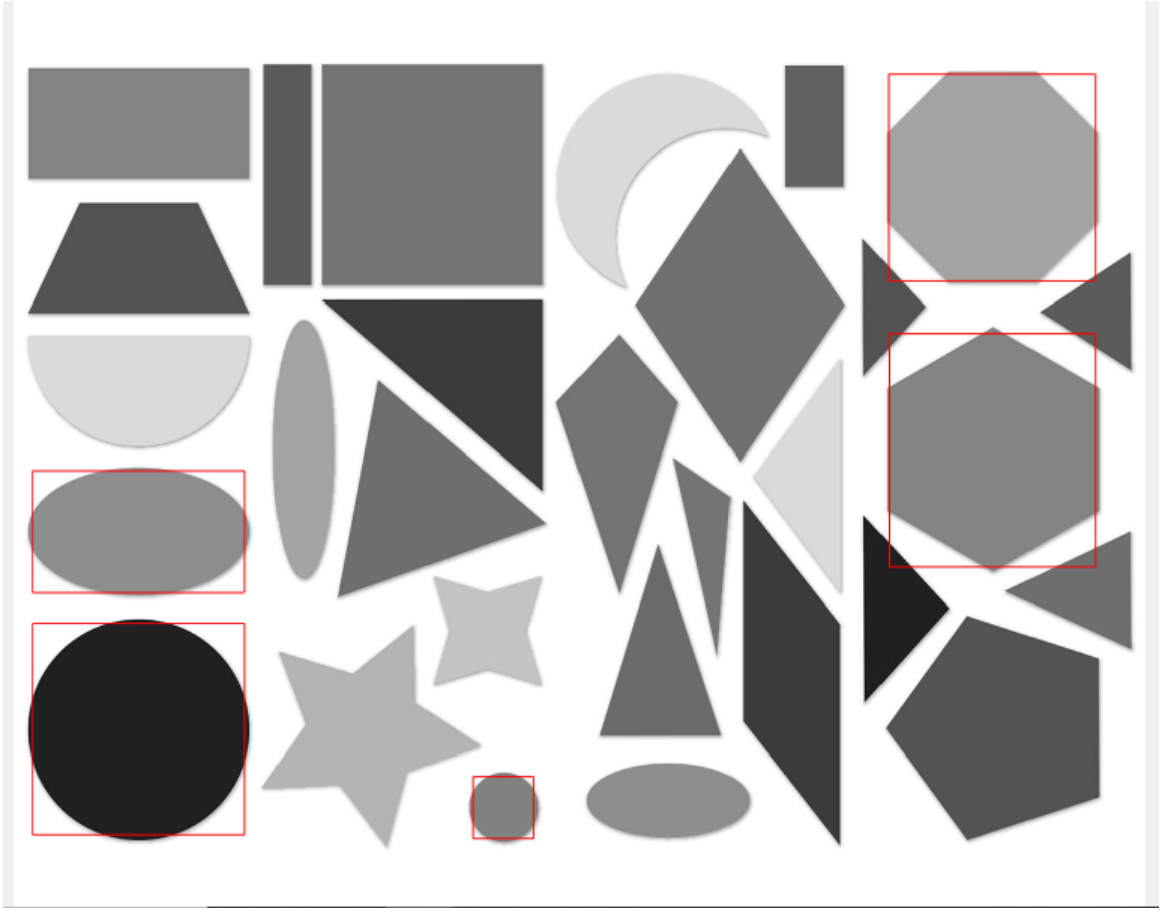
Figure 5



3.2 Geomatrical shapes

Figure 1

Figure 2



4 Discussion

regionprops() is an important tool to find the properties of the segmented objects within an image. There are a lot of segmentation properties in the regionprops documentation in Matlab which can help us in various classification of image applications in future. We used circularity as one of the properties in the section Geomatrical shapes.

5 Appendix

5.1 FLIR image

```
set(0, 'defaulttextinterpreter','Latex');
IR = imread('FLIR0163.jpg');
```

```
RGB = imread('FLIR0163- photo.jpg');

figure(1),imshow(IR),title('IR image'),impixelinfo;
figure(2),imshow(RGB),title('RGB image'),impixelinfo;

i = RGB(:,:,1);  %red band
j = IR(:,:,1);

figure(3),
subplot(2,1,1),imhist(j),title('IR histogram (red band
    )');
subplot(2,1,2),imhist(i),title('RGB histogram (red
    band)');

%% Supervised classification
V = double([i(:),j(:)]);
g = [1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4]';    %
    group number
t = [ 2 205                                          %
    pixel values
       4 211
      31 214
      25 202
      14 227
       2 236
      10 233
      27 231
      29 233
       2 240
       4 226
      30 230
      27 247
       5 227
       2 245
       4 234
      29 240
       4 214
       6 243
      27 212];

class = classify(V, t, g);
a = vec2mat(class, size(i,1));
b = ind2rgb(a', prism);
figure(4),imshow(b),title('Supervised'),impixelinfo;
%% regionprops()
```

```
BW=im2bw(b);
L = bwlabel(BW,8);
regionMeasurements = regionprops(L, 'basic');
allAreas = [regionMeasurements.Area];

[sortedAreas, sortIndexes] = sort(allAreas, 'descend')
    ;

biggestArea1 = sortedAreas(1:1);

new_Biggest=(L==sortIndexes(1));
double(new_Biggest);
imBiggest=double(BW).*new_Biggest;

figure(5),imshow(imBiggest,[]),title('Object in
    question'),impixelinfo;
```

5.2 Geomatrical shapes

```
set(0, 'defaulttextinterpreter','Latex');
im=imread('geom_shapes.jpg');
im=rgb2gray(im);
E = entropyfilt(im);  %texture image
Eim = rescale(E);
bwim=imbinarize(Eim);
bg = (~bwim);   %background

figure(1),
subplot(2,1,1),imshow(bwim),title('Foreground');
subplot(2,1,2),imshow(bg),title('Background'),
    impixelinfo();

[labeled, num]=bwlabel(bg,8);
regions=regionprops(labeled,'all');
regionSize = size(regions);

allPerimeters = [regions.Perimeter];
allAreas = [regions.Area];

allCircularities = allPerimeters.^ 2./(4*pi*allAreas);
[sortedCircularities, circleIndexes] = sort(
    allCircularities, 'descend');

matchIndex = 1;
for i = 1:num
    if  allCircularities(i) < 1.19
```

```
        matches(matchIndex) = regions(i);
        matchIndex =matchIndex + 1;

        figure(2),imshow(im),
        hold on
        for j=1:length(matches)
            rectangle('Position',matches(j).
               BoundingBox,'EdgeColor','r')
        end
    end
end
circularobjects = matchIndex-1;
disp(circularobjects)
```