Image Processing

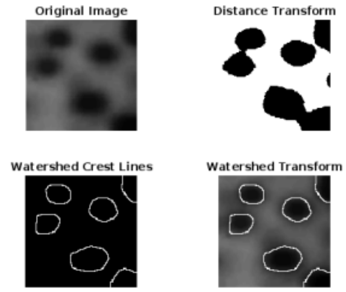
by assist Assist

Submission date: 07-Nov-2022 04:44AM (UTC-0600)
Submission ID: 1947020785
File name: Image_processing_in_MATLA1.docx (668.18K)
Word count: 1472
Character count: 10416

Image Processing in MATLAB

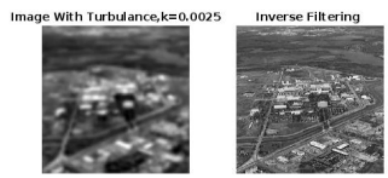
1)
Edge detection is practically a part of processing of image that is fundamental. This specifically helps to identify digital image points where the brightness is sharp. These specific points that set the curve are known as edges. Implementation of edge detection have different methods, one of which is based on zero-crossing that is particularly known as edge detection filter of Marr-Hildreth [1].

```
clear all;
close all
orgImg=imread('figBlob.tif');
orgImg=im2double(orgImg);
figure,subplot(221);imshow(orgImg);title('Original Image');
%binary map of image by thresholding with otsu's method
bw=~imbinarize(orgImg,graythresh(orgImg));
%distance transform of image
D=bwdist(bw);
subplot(222);imshow(D);title('Distance Transform');
%watershed
%gradient image
h= fspecial('prewitt');
gx=imfilter(orgImg, h, 'replicate');
gy= imfilter(orgImg, h', 'replicate');
gm=sqrt(gx.^2+gy.^2);
%smoothening
g=imfilter(gm, fspecial('disk',4));
%binary image & distance transform
bw1=imbinarize(g, graythresh(g));
D1=bwdist(~bw1);
L = watershed(D1);
%watershed crest lines
w = (L==0);
%watershed boundary overlappped to original
wsSeg=orgImg;
wsSeg(w)=1;
subplot(223);imshow(w);title('Watershed Crest Lines');
subplot(224);imshow(wsSeg,[]);title('Watershed Transform');
```



For inverse filtering

```
clear all;
close all;
img=imread('aerial_view_no_turb.tif');
[M,N]=size(img);
FM=fftshift(fft2(img));
k=0.0025; %turbulance
for i=1:M
  for j=1:N
    H(i,j)=exp(-k*((i-M/2)^2+(j-N/2)^2));
  end
end
GM=FM.*H;
noimg=ifft2(GM);
figure,subplot(1,2,1),imshow(abs(noimg),[]);title(['Image With Turbulance,k=',num2str(k)])

%Inverse Filtering
OUT=GM./H;
inv=ifft2(OUT);
subplot(1,2,2),imshow(abs(inv),[]),title('Inverse Filtering ');
```



2)
Methods of Otsu’s threshold involves iterating by the possible means of values of threshold and measure of spread calculation for pixel levels for each threshold side, this implies, pixels either fall in background or foreground [2].

```
clear all
close all
```

```
im=imread('blob_original.tif');
figure,subplot(131),imshow(im)
title('Original Image')
T=60;
first=(im<=T);
second=(im>T);
seg=1*first+255*second;
subplot(132),imshow(uint8(seg))
title(['Thresholded Image, T=',num2str(T)])

%Otsu method
level = graythresh(im);
BW = im2bw(im,level);
subplot(133),imshow(BW)
title('Otsu Method')
```

5

```
clear all
close all
orgImg=imread('figBlob.tif');
f=imadjust(orgImg, [0 1], [1 0]);
se=strel('disk',20,8);
S=imerode(f, se);
T=input('Enter the Threshold: ');
if numel(S) == 1
   SI = f == S;
   S1 = S;
else
% S is an array. Eliminate duplicate, connected seed locations
% to reduce the number of loop executions in the following
% sections of code.
   SI = bwmorph(S, 'shrink', Inf);
   S1 = f(SI(:)); % Array of seed values.
end
TI = false(size(f));
for K = 1:length(S1)
   seedvalue = S1(K);
   S2 = abs(f - seedvalue) <= T;%predicate
   TI = TI | S2;
end
% Use function imreconstruct with SI as the marker image to
% obtain the regions corresponding to each seed in S. Function
% bwlabel assigns a different integer to each connected region.
[g,NR] = bwlabel(imreconstruct(SI, TI));
subplot(121);imshow(orgImg);title('Original Image');
subplot(122);imshow(1-g);title('Segmentation based on Region growing');
```

3)

The application of edge detector to take input image that is unfiltered. The application of the input image is unfiltered. Thus, the side by side filtered images are displayed as comparison. This function is practically used for segmentation of image and extraction of data in areas of processing of image, machine vision and computer vision. Detection algorithm for common edge include Canny, Sobel, Roberts, Prewitt and methods of fuzzy logic [3].

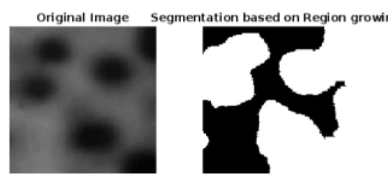
```
function splitAndMerge()
   orgImg=imread('figXray.tif');
   md=input('Enter the minimum dimension of block: ');

segImg=splitmerge(orgImg,md,@predicate);
   subplot(121);imshow(orgImg);title('Original Image');

subplot(122);imshow(segImg);title('Segmented Image');
end

function g = splitmerge(f,mindim,fun)
   q=2^nextpow2(max(size(f)));
   [row,col]=size(f);
   f=padarray(f,[q-row,q-col],'post');
   z=qtdecomp(f,@split_test,mindim,fun);
   Lmax=full(max(z(:)));
   g=zeros(size(f));
   marker=zeros(size(f));
   for k=1:Lmax
      [vals,r,c]=qtgetblk(f,z,k);
      if ~isempty(vals)
         for i=1:length(r)
            xlow=r(i);
            ylow=c(i);
            xhigh=xlow+k-1;
            yhigh=ylow+k-1;
            region = f(xlow:xhigh,ylow:yhigh);
            flag=fun(region);
            if flag
               g(xlow:xhigh,ylow:yhigh)=1;
               marker(xlow,ylow)=1;
            end
         end
```

```
        end
    end
    g=bwlabel(imreconstruct(marker,g));
    g=g(1:row,1:col);
end

function flag=predicate(region)
    sd=std2(region);
    m=mean2(region);
    flag=(sd>10) & (m>0) & (m<125);
end

function v=split_test(b,mindim,fun)
    k=size(b,3);
    v(1:k)=false;
    for i=1:k
        quadregion=b(:,:,i);
        if size(quadregion,1)<=mindim
                v(i)=false;
        continue
        end
            flag=fun(quadregion);
        if flag
                v(i)=true;
        end
    end
end
```

4)

Several methods that include histogram method, method that is entropy based, method that uses busyness measures, etc. are commonly known as problems of image segmentation for threshold selection. This particular implementation is based on several measures of entropy on both colour and grayscale images. Measures of entropy are particularly matched better with Otsy method [4].

```
clear all
close all
orgImg=imread('figBlob.tif');
f=imadjust(orgImg, [0 1], [1 0]);
se=strel('disk',20,8);
S=imerode(f, se);
T=input('Enter the Threshold: ');
if numel(S) == 1
   SI = f == S;
   S1 = S;
else
% S is an array. Eliminate duplicate, connected seed locations
% to reduce the number of loop executions in the following
% sections of code.
   SI = bwmorph(S, 'shrink', Inf);
   S1 = f(SI(:)); % Array of seed values.
end
TI = false(size(f));
for K = 1:length(S1)
   seedvalue = S1(K);
   S2 = abs(f - seedvalue) <= T;%predicate
   TI = TI | S2;
end
% Use function imreconstruct with SI as the marker image to
% obtain the regions corresponding to each seed in S. Function
% bwlabel assigns a different integer to each connected region.
[g, NR] = bwlabel(imreconstruct(SI, TI));
subplot(121);imshow(orgImg);title('Original Image');
subplot(122);imshow(1-g);title('Segmentation based on Region growing');
```



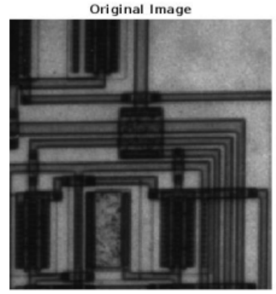
Histogram equalised intensity

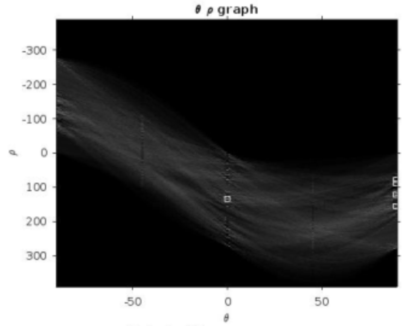
```
clear all
close all
im=imread('lena_RGB.tif');
[h,s,i]=rgb2hsv(im);
ih=histeq(i);
new(:,:,1)=h;
new(:,:,2)=s;
new(:,:,3)=ih;
newim=hsv2rgb(new);
subplot(2,2,1),imshow(im);title('RGB Image');
subplot(2,2,2),imshow(i);title('Original Intensity ');
subplot(2,2,3),imshow(ih);title('Histogram Equalized Intensity');
subplot(2,2,4),imshow(newim);title('New RGB image');
```

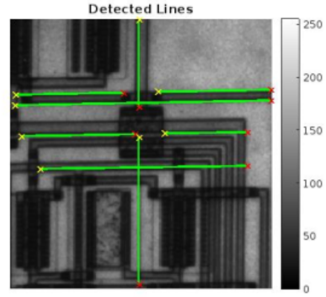
5)

The techniques of thresholding that involves Entropy-based for local and joint methods of entropy to make sense of focused matching between images while later is particularly introduced to emphasise on one image's co-occurrence matrix. These techniques are particularly image dependent and it happens to be that the relative and local entropy seem to work better than joint entropy [5].

```
clear all;
close all;
I  = imread('circuit.tif');
figure,imshow(I),title('Original Image')
BW = edge(I,'canny');
[H,theta,rho] = hough(BW);
figure,imshow(H,[],'XData',theta,'YData',rho,'I
nitialMagnification','fit');
title('\theta \rho graph')
xlabel('\theta'), ylabel('\rho');
axis on, axis normal, hold on;
P = houghpeaks(H,5);
x = theta(P(:,2));
y = rho(P(:,1));
plot(x,y,'s','color','white');
lines                                          =
houghlines(BW,theta,rho,P,'FillGap',20,'MinL
ength',40);
figure, imshow(I), title('Detected Lines'),hold
on
max_len = 0;
for k = 1:length(lines)
   xy = [lines(k).point1; lines(k).point2];

plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green
');

   % Plot beginnings and ends of lines

plot(xy(1,1),xy(1,2),'x','LineWidth',2,'Color','y
ellow');

plot(xy(2,1),xy(2,2),'x','LineWidth',2,'Color','re
d');
end
```







Hue, saturation & intensity

```
clear all
close all
im=imread('lena_RGB.tif');
R=im2double(im(:, :, 1));
```

```
G=im2double(im(:, :, 2));
B=im2double(im(:, :, 3));
[M, N]=size(R);

% Hue calculation
HUE=acosd((0.5*double((R-G)+(R-B)))./(sqrt(double(((R-G).^2)+((R-B).*(G-B))))));
HUE(B>G)=360-HUE(B>G);

% Saturation
SAT= (ones(M, N))-((3.*min(R,min(G,B)))./(R+G+B));

% Intensity
INT=(R+B+G)/3;

figure
subplot(221); imshow(im); title('Original image');
subplot(222); imshow(HUE,[]); title('Hue image');
subplot(223); imshow(SAT); title('Saturation image');
subplot(224); imshow(INT); title('Intensity image');
```

6)

For implementation of local, relative and joint entropy it seems that, relative entropy could practically compliment the local and the joint entropy in terms for provisioning for different details that others are not able to. Considering computing saving, the approach of relative entropy also specifically provides computational complexity that is the least [6].

```
clc; clear all; close all;
orgImg = im2double(imread('cameraman.tif'));
[r,c] = size(orgImg);
h = fspecial('motion',20,45);
g = imfilter(orgImg,h,'circular');
n = imnoise(zeros(r,c),'gaussian');
size(n);
size(g);
g = g + n;

subplot(131); imshow(g); title('Degraded Image');
% Sn(u,v) = |N(u,v)|^2 = Power Spectrum of Noise
Sn = abs(fft2(n)).^2;
% Sf(u,v) = |F(u,v)|^2 = Power Spectrum of Image
Sf = abs(fft2(orgImg)).^2;
% Average Noise Power = Sum(Sn)/(r*c);
nA = sum(Sn(:))/numel(Sn);
% Average Image Power = Sum(Sn)/(r*c);
fA = sum(Sf(:))/numel(Sf);
R = nA/fA;
fcap=deconvwnr(g, h, R);
subplot(132); imshow(fcap); title('Constant Ratio');
% Noise correlation
NCORR = fftshift(real(ifft2(Sn)));
%Image correlation
ICORR = fftshift(real(ifft2(Sf)));
Auto=autocorr(g(:));
fcap2 =deconvwnr(g, h, NCORR,ICORR);
subplot(133); imshow(fcap2);
title('Autocorrelation');
```

7)

A GUI for evaluating the functions that are integral by f(x) function. It could be used by GUI at the command prompt by entering ‘nu’.

```
function varargout = ImageM(varargin)
% Run a new instance of ImageM application.
%
%   ImageM
%   Creates a new ImageM window, with a menu and without image.
%
%   ImageM(IMG)
%   Creates a new ImageM window initialized with the given image. IMG
%   should be an instance of Image Object.
%
%   VIEWER = ImageM(IMG);
%   Returns the ImageM Viewer object created for the input image.
%   The viewer contains several fields, among them:
%   * Gui:  the global GUI that manages the set of frames/viewers
%   * Doc:  an ImagemDoc object that encapsulates the image together with
%          useful information
```

```
%   * Handles: a set of handles to the widgets that constitute this viewer.
%
%
%   Example
%     img = Image.read('cameraman.tif');
%     ImageM(img);
%
%   See also
%         Image, imagem.app.ImagemAppData, imagem.gui.ImagemGUI
%

% check if image is present, or create one
img = [];
if ~isempty(varargin)
    var = varargin{1};

    if isa(var, 'Image')
        % if first argument is an image object, keep it
        img = var;

        % if image has no name, use the name of the variable
        if isempty(img.Name)
            img.Name = inputname(1);
        end

    elseif ischar(var)
        % if first input is a string, use it to open an image
        img = Image.read(var);

    elseif isnumeric(var) || islogical(var)
        % if input is numerical array, convert to image and keep input name
        img = Image(var);
        img.Name = inputname(1);
    end
end

% create the application, and a GUI
app = imagem.app.ImagemAppData;
gui = imagem.gui.ImagemGUI(app);

% use the GUI to create a new image display
viewer = createImageFrame(gui, img);


% returns handle to viewer if requested
if nargout > 0
    varargout = {viewer};
end
```

References

[1] e. a. Sujithra.B.S, "Eminent Identification And Segmentation Of Optic Disk In Digital Fundus Images Using Marr-Hildreth Operator," vol. volume 12, 2021.

[2] A. Akagic, "Pavement crack detection using Otsu thresholding for image segmentation," 2018.

[3] T. Zhu, "Generalized Spatial Differentiation from the Spin Hall Effect of Light and Its Application in Image Processing of Edge Detection," 2019.

[4] Y. Liu, "Entropy-Based Image Fusion with Joint Sparse Representation and Rolling Guidance Filter," 2020.

[5] M. Chouksey, "A Joint Entropy for Image Segmentation Based on Quasi Opposite Multiverse Optimization," 2021.

[6] D. Minnen, "Channel-Wise Autoregressive Entropy Models for Learned Image Compression," 2020.

Image Processing

ORIGINALITY REPORT

41%	37%	15%	35%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	archive.org Internet Source	11%
2	diptutor.wordpress.com Internet Source	5%
3	Submitted to University of Queensland Student Paper	4%
4	Submitted to CSU, Fullerton Student Paper	3%
5	www.nbhkdz.com Internet Source	3%
6	Submitted to University of Greenwich Student Paper	2%
7	Submitted to Wright State University Student Paper	2%
8	Submitted to University of South Australia Student Paper	2%
9	Submitted to Monash University Student Paper	2%

10	bbs.csdn.net Internet Source	1 %
11	Submitted to Utah Valley State College Student Paper	1 %
12	(11-23-12) http://202.197.67.17/yxtxcl/syzd/doc/%E5%AE%9E%E9% Internet Source	1 %
13	Submitted to University of Southern California Student Paper	1 %
14	Submitted to Universiti Malaysia Pahang Student Paper	1 %
15	dokumen.pub Internet Source	1 %
16	orca.cf.ac.uk Internet Source	1 %
17	www.slideshare.net Internet Source	1 %
18	docplayer.gr Internet Source	<1 %
19	faculty.kutztown.edu Internet Source	<1 %
20	vision.technion.ac.il Internet Source	<1 %