EECE-5626 (IP&PR) : Homework-1 Solutions

Table of Contents

Problem 1.1: DIP4E Problem 2.10

An automobile manufacturer is automating the placement of certain components on the bumpers of a limited-edition line of sports cars. The components are color-coordinated, so the assembly robots need to know the color of each car in order to select the appropriate bumper component. Models come in only four colors: blue, green, red, and white. You are hired to propose a solution based on imaging. How would you solve the problem of determining the color of each car, keeping in mind that cost is the most important consideration in your choice of components?

Solution: One possible solution is to equip a monochrome camera with a mechanical device that sequentially places a red, a green, and a blue pass filter in front of the lens. The strongest camera response determines the color. If all three responses are approximately equal, the object is white. A faster system would utilize three different cameras, each equipped with an individual filter. The analysis then would be based on polling the response of each camera. This system would be a little more expensive, but it would be faster and more reliable.

Yet another alternative would be to use a color camera directly, and then use software algorithms to determine the color being seen by the camera. In order to determine if this solution is cheaper than the the 3-camera approach we would have to know the cost of the hardware and software needed to implement the detection algorithm.

Note that all solutions assume that the field of view of the camera(s) is such that it is completely filled-in by a uniform color [i.e., the camera(s) is (are) focused on a part of the vehicle where only its color is seen. Otherwise further analysis would be required to isolate the region of uniform color, which is all that is of interest in solving this problem. Illumination should be such that there are no reflections, so that all images are uniform to make color detection easier and faster (and less expensive).

Problem 1.2: DIP4E Problem 2.13

When discussing linear indexing in Section 2.4, we arrived at the linear index in Eq. (2.14) by inspection. The same argument used there can be extended to 2 3-D array with coordinates x, y, and z, and corresponding dimensions M, N, and P. The linear index for any (x, y, z) is

$$s = x + M(y + Nz). \qquad (2.1)$$

Start with this expression and derive the following.

(a) Eq. (2-15) for 3-D array.

Solution: The mod operation '$\alpha \bmod M$' means "the remainder of the division of α by M." From (2.1) we have $s = x + My + MNz$. Then a straightforward approach is to perform first $\bmod$ operation with respect to MN to eliminate z and then perform second $\bmod$ operation on the result with respect to M to eliminate y, that is

$$x = \big((s \bmod MN) \bmod M\big).$$

However, in Eq. (2.1) note that $(y + Nz)$ is an integer. Then one $\bmod$ operation on s with respect to M will eliminate both y and z and the remainder will be x, or

$$x = (s \bmod M).$$

Thus, the DIP4E equation $(2\text{-}15)$ is still valid for 3-D array.

MATLAB Check:

```
clc; close all; clear;
M = 4; N = 3; P = 2; MNP = M*N*P;
s = 0:MNP-1; format short; s = reshape(s,M,N,P)
```

```
s =
s(:,:,1) =
     0     4     8
     1     5     9
     2     6    10
     3     7    11
s(:,:,2) =
    12    16    20
```

```
    13    17    21
    14    18    22
    15    19    23
```

```
x1 = mod(mod(s,M*N),M), % straightforward approach
```

```
x1 =
x1(:,:,1) =
     0     0     0
     1     1     1
     2     2     2
     3     3     3
x1(:,:,2) =
     0     0     0
     1     1     1
     2     2     2
     3     3     3
```

```
x2 = mod(s,M), % efficient approach
```

```
x2 =
x2(:,:,1) =
     0     0     0
     1     1     1
     2     2     2
     3     3     3
x2(:,:,2) =
     0     0     0
     1     1     1
     2     2     2
     3     3     3
```

This shows that $x = s \bmod M$ is sufficient.

(b) Eq. (2-16) for 3-D array.

Solution: From Eq. (2.1), we have $M(y + Nz) = s - x$ and

$$y + Nz = (s - x)/M. \qquad (2.2)$$

To eliminate z from the left-hand-side, we perform the "$\bmod$" operation with respect to N on both sides of (2.2) to obtain

$$y = \big([(s - x)/M] \bmod N\big). \qquad (2.3)$$

However, the DIP4E equation $(2\text{-}16)$ is $y = (s - x)/M$ is not valid for 3-D arrays.

MATLAB check:

```
y1 = mod((s-x1)/M,N), % Correct approach
```

```
y1 =
y1(:,:,1) =
     0     1     2
     0     1     2
     0     1     2
     0     1     2
y1(:,:,2) =
     0     1     2
     0     1     2
     0     1     2
```

```
     0     1     2
```

```
y2 = (s-x1)/M, % DIP4E Eq. (2-16)
```

```
y2 =
y2(:,:,1) =
     0     1     2
     0     1     2
     0     1     2
     0     1     2
y2(:,:,2) =
     3     4     5
     3     4     5
     3     4     5
     3     4     5
```

Clearly, $y = (s - x)/M$ in DIP4E equation $(2\text{-}16)$ is incorrect.

Problem 1.3: DIP4E Problem 2.52

Let $\{z_k\}$, $k = 1, 2, \ldots, K$, be a set of samples representing an entire population. In the following, assume that $p(z)$ is estimated using Eq. (2.89) or $p(z) = \frac{h(z)}{K}$ where $h(z)$ is the histogram or the number of times that value z occurs in the K observations.

(a) Show that the same results for the mean are obtained using Eq. (2.92) or (2.110).

Solution: From Eq. (2.92), we have

$$\bar{z} = E[z] = \sum_{z \in R} z\, p(z)$$

where R is the range of z. Let us assume that there are Q distinct values $\{r_q\}_{q=1}^{Q}$ in R and hence in $\{z_k\}$, that is,

$$R = \{r_1, r_2, \ldots, r_Q\}.$$

Note that this assumption is appropriate for images since they have integer values. The elements of $\{z_k\}$ can be grouped in Q groups, where each group contains the elements of $\{z_i\}$ that have the same value. Then the histogram $h(z)$ is given by

$$h(z) : \{h(1), h(2), \ldots, h(Q)\}, \quad h(1) + h(2) + \cdots + h(Q) = K$$

where $h(q) =$ number of elements in group r_q. Hence, the estimated probability mass function (PMF) is given by

$$p(z) = \frac{h(z)}{K} = \frac{\{h(1), h(2), \ldots, h(Q)\}}{K}.$$

Consider Eq. (2.100) to estimate the population mean $\bar{z}$:

$$\begin{aligned}\bar{z} &= \frac{1}{K}\sum_{k=1}^{K} z_k = \frac{1}{K}\left[r_1 h(1) + r_2 h(2) + \cdots + r_Q h(Q)\right] \\ &= \frac{1}{K}\sum_{q=1}^{Q} r_q h(q) = \sum_{q=1}^{Q} r_q \frac{h(q)}{K} = \sum_{z\in R} z\frac{h(z)}{K} = \sum_{z in K} z\, p(z)\end{aligned}$$

which is the same result as in Eq. (2.92).

(b) Show that the same result for the variance is obtained using Eq. (2.93) or (2.111).

Solution: The solution to this part is an extension of the solution to (a), only the form of the grouping changes:

$$\frac{1}{K}\sum_{k=1}^{K}\left(z_k - \bar{z}\right)^2 = \frac{1}{K}\left[\left(r_1 - \bar{z}\right)^2 h(1) + \cdots + \left(r_Q - \bar{z}\right)^2 h(Q)\right].$$

Following the same argument as in part (a), we can show that

$$\frac{1}{K}\left[(r_1 - \bar{z})^2 h(1) + \cdots + (r_Q - \bar{z})^2 h(Q)\right] = \sum_{z\in R}(r - \bar{z})^2 p(z)$$

which proves that

$$\frac{1}{K}\sum_{k=1}^{K}(z_k - \bar{z})^2 = \sum_{z\in R}(r - \bar{z})^2 p(z)$$

so the same result is obtained for the variance using either Eq. (2.93) or (2.111).

(c) Show that the nth central moment using Eq. (2.96) or (2.112) gives the same result.

Solution: The proof is identical to part (b) by replacing $(z_k - \bar{z})^2$ by $(z - \bar{z})^n$.

Problem-1.4: DIP4E Project 2.10 (Statistical Central Moments)

(a) Function `centralMoments4e`

```
function u = centralMoments4e(f,n)
% CENTRALMOMENTS4E Computes centrai moments.
% U = CENTRALMOMENTS4E(F,N) Computes the N central moments of the
% intensities of an 8-bit grayscaie image F, with intensities in
% the range [0,1] or [0,255]. The results are output in
% N-dimensionai vector U such that U(1) is the mean, U(2) the
% variance, U(3) the third central moment, and so on. Moment
% computations are based on Eq. (2-96).

% Copyright 2017, R. C. Gonzaiez & R. E. woods

%-Compute the normalized histogram of F using project function
% imageHist4e.
p = imageHist4e(f);

%—Detennine if the intensities of F are in the range [0,1] or
```

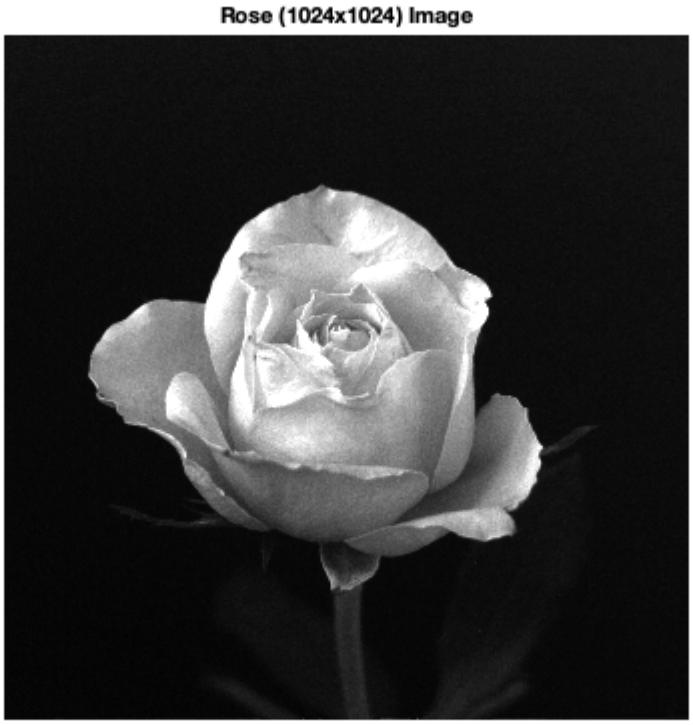
```
% [0,255]. Need to do this in order to define the vaiues of 2 needed
% to compute the moments.

if max(f(:)) <= 1
    de1 = 1/255;
    z = 0:de1:1;
else
    de1 = 1;
    z = 0:de1:255;
end

%-Prea11ocate memory for 1oop speed.
u(1:n) = 0;

%-Compute the mean.
m = sum(z.*p); u(1) = m;

%-Compute the rest of the moments.
for I = 2:n
    u(I) = sum(((z - m).^I).*p);
end
end
```

(b) Moment Computation

Compute the mean, variance, third, and fourth moments of the image rose1024.tif. Display the image and the moments.

Solution: MATLAB script

```
clc; close all; clear;
f1 = imread('../artfiles/rose1024.tif');
figure('units','inches','position',[0,0,8,4]);
imshow(f1(1:3:end,1:3:end));
title('Rose (1024x1024) Image');
```



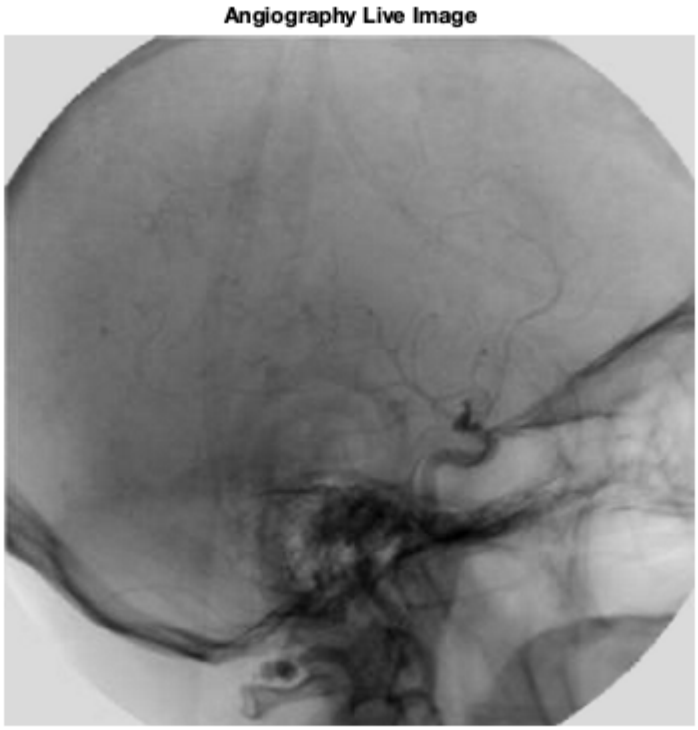

```
u1 = centralMoments4e(f1,4); format short eng; display(u1);
```

```
u1 = 1×4
    46.7894e+000     4.3698e+003   503.5215e+003    89.4575e+006
```

(c) Repeat (b) for the image `angiography-live-image.tif`.

Solution: MATLAB script

```
f2 = imread('../artfiles/angiography-live-image.tif');
figure('units','inches','position',[0,0,3,3]);
imshow(f2(1:2:end,1:2:end));
title('Angiography Live Image');
```



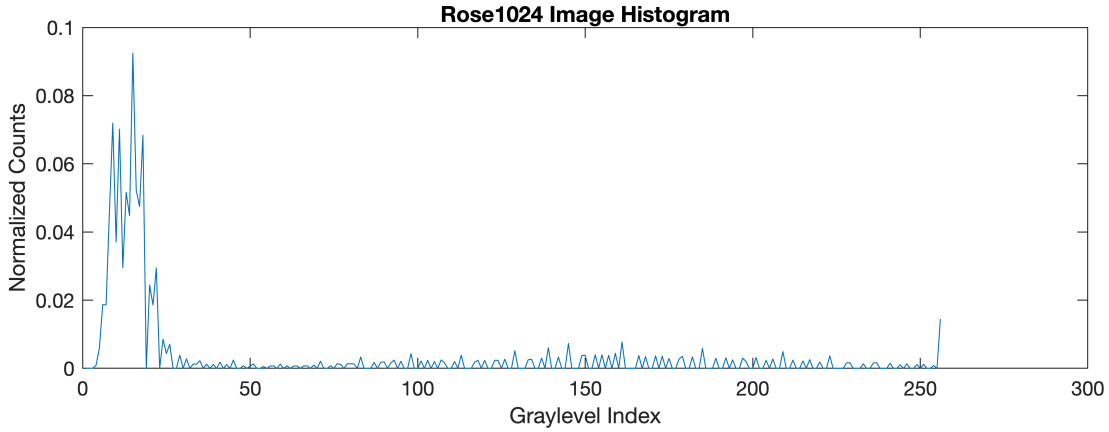

```
u2 = centralMoments4e(f2,4); format short eng; display(u2);
```

```
u2 = 1×4
   129.3007e+000     1.8572e+003    14.9847e+003    10.2903e+006
```

(d) Compute and plot the histograms of both images.

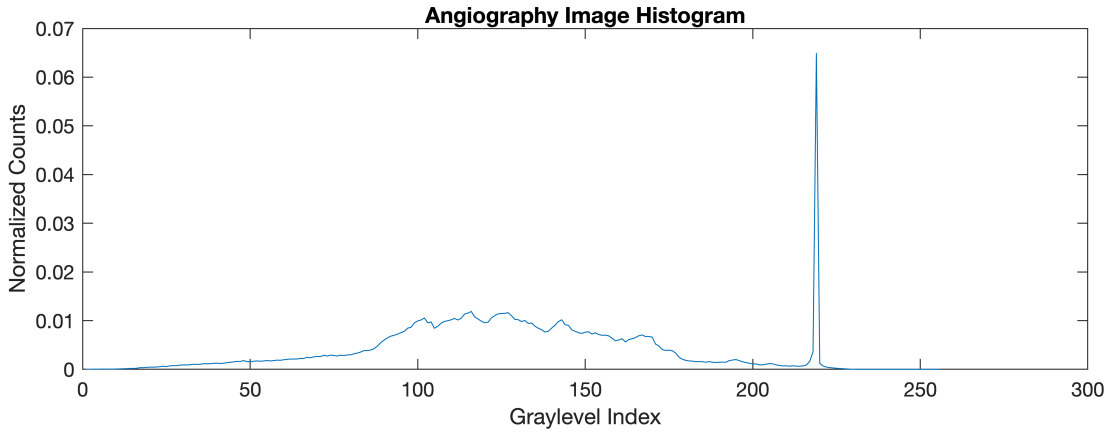
Solution: MATLAB script

```
h1 = imageHist4e(f1); h2 = imageHist4e(f2);
figure('units','inches','position',[0,0,9,3]);
plot(h1); xlabel('Graylevel Index'); ylabel('Normalized Counts');
title('Rose1024 Image Histogram');
```




```
plot(h2); xlabel('Graylevel Index'); ylabel('Normalized Counts');
```

```
title('Angiography Image Histogram');
```



(e) Discuss how the values obtained in (b) and (c) relate to the shape of the histograms in (d).

Discussion: The mean of `f2` is higher, indicating a lighter image. As you can see, the histogram of this image has a significantly higher number of pixels biased to the right, as compared with the histogram of image `f1`. The variance of `f1` is considerably higher, indicating a higher contrast image. The histogram of `f1` has dominant peaks at the low and high ends of the intensity scale, making the image look almost binary, thus the higher contrast. The histogram of `f2` is more symmetrical than the histogram of `f1` (the peak on the right of the former is narrow, and does not have as much influence on the third moment as the peaks in the histogram of `f1`), thus we would expect the third moment to be smaller for `f2` than for `f1`. In terms of the fourth moment, the histogram of `f1` is flatter than the other histogram because of the "hump" in the middle of the histogram of `f2`; thus, we would expect the fourth moment to be larger for the histogram of `f1`.

Problem 1.5: DIPUM3E MATLAB Projects 2.3 and 2.6

Image generation using **`iswholeTest`** and **`testImage`** functions.

(a) MATLAB Project 2.3

Function **`iswholeTest`**

```
function [W,S] = iswholeTest(A)
% iswho1eTest: Elements of an array that are whole numbers (integers)
% [W, S] = iswholeTest(A) returns a logical array, w, of the same size
% as A, with 1s (TRUE) in the locations corresponding to whole numbers
% in A, and 0s (FALSE) elsewhere. Scalars is 1 (TRUE) if ALL elements
% of A are whole numbers; otherwise S is 0 (FALSE). A must be a real,
% finite, numeric array. S is used in cases where testing the entire
% array as a unit is required. Note: For complex arrays, apply this
% function to the real and imaginary components separately.

% Check the validity of A.
if ~isnumeric(A) || ~isreal(A) || ~isfinite(A)
    error('A must be a real, finite, numeric array.')
end
```

```
% If the e1ements of A are f1oating point numbers, detect which of them
% are whole numbers.
if isa(A, 'double' ) || isa(A, 'single')
    W = A == floor(A);
else
    % The elements of A must be one of the MATLAB integer types, so all
    % the elements of I are true (1).
    W = true(size(A));
end

% see if all elements of W are 1 (TRUE):
S = false;
if sum(W(:)) == numel(W)
    S = not(S);
end
end
```

(b) MATLAB Project 2.6

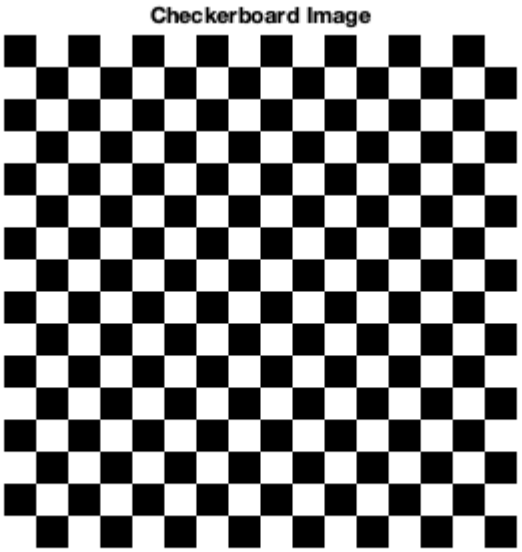
(i) MATLAB function `testImage`

```
function f = testimage(type,param)
%testimage Generates gray scale test images.
% F = testrmage(TYPE, PARAM) generates a grayscale image of size M-by-N
% and class double. TYPE is a string, and PARAM is a vector of
% parameters, as described below:
%
%      TYPE       PARAMETERS               DESCRIPTION
%  'checkerboard'  [M,N,n]        checkerboard image of
%                                 size M-by-N, with each square of size
%                                 n-by-n pixels. Note: Both M N must be
%                                 divisibe by 2n; otherwise the
%                                 checkerboard will not make sense.
%  'unoise'         [M,N]         Image of size M-by-N whose pixels
%                                 are uniformly distributed random
%                                 numbers in the range [0,1].
%  'gnoise'         [M,N]         Image of size M-by-N whose pixels
%                                 are random numbers drawn from a
%                                 standard normal distribution (i.e., a
%                                 normal(Gaussian) distribution with
%                                 mean of 0 and standard deviation
%                                 equal to 1).
%  'vstripe'        [M,N,n]       M-by-N black image with a white,
%                                 centered, vert stripe n pixels wide.
%                                 n cannot exceed N.
%  'hstripe'        [M,N,n]       M-by-N black image with a white,
%                                 centered, horiz stripe n pixels wide.
%                                 n cannot exceed M.


% Preliminaries.
M = param(1); N = param(2);
if numel(param) == 3
    n = param(3);
end
if isequal(type,'checkerboard') % M and N must be divisible by 2n.
    if ~iswholeTest(M/(2*n)) || ~iswholeTest(N/(2*n))
        error('For checkerboard, Mand N must be divisible by 2*n')
```

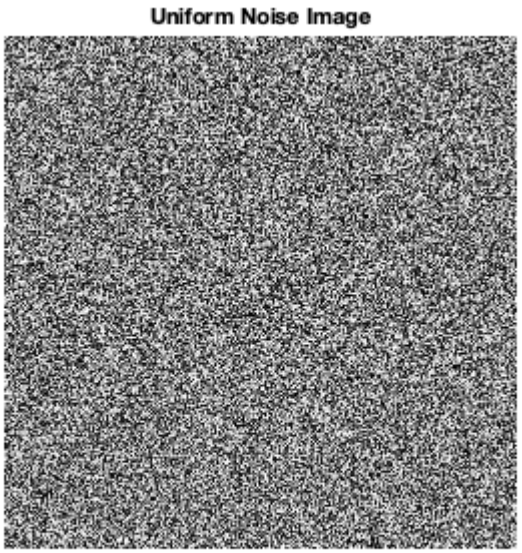
```
    end
elseif (isequal(type, 'vstripe') && n > N) ...
        || (isequal(type, 'hstripe') && n > M)
    error('The stripe is wider than one image dimension')
end

f = 1;
% Generate image specified in TYPE.
switch type
    case 'checkerboard'
        % Have to compute checkeboard parameters so that the
        % resulting image will be of size M-by-N.
        p = M/(2*n);
        q = N/(2*n);
        f = checkerboard(n,p,q) > 0.5; % convert default gray to white.
        % The output of function checkerboard is logical. convert to
        % double.
        f = im2double(f);
    case 'unoise'
        f = rand(M,N);
    case 'gnoise'
        f = randn(M,N);
    case 'vstri pe'
        f = zeros(M,N);
        cy = floor(N/2); % vertical center.
        lowlim = cy - floor(n/2) + 1;
        highlim = lowlim + n - 1;
        f(:, lowlim:highlim) = 1;
    case 'hstripe'
        f = zeros(M,N);
        ex = floor(M/2); % Horizontal center.
        lowlim = ex - floor(n/2) + 1;
        highlim = lowlim + n - 1;
        f(lowlim:highlim, :) = 1;
end

end
```

(ii) Image Generation and display

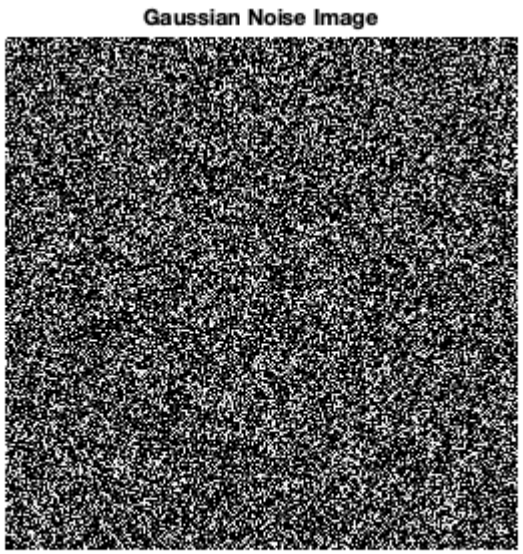
```
clc; close all; clear;
% Image size
M = 512; N = 512;
% Checkerboard image
fcheck = testImage('checkerboard',[M,N,32]);
figure; imshow(fcheck(1:2:M,1:2:N));
title('Checkerboard Image');
```



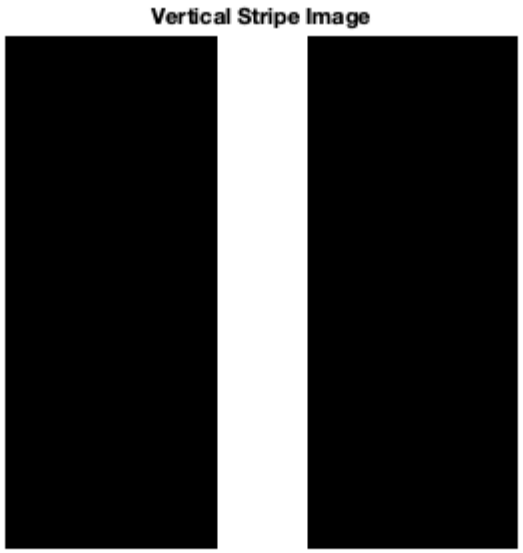

```
% Uniform noise image
fu = testImage('unoise',[M,N]);
figure; imshow(fu(1:2:M,1:2:N));
title('Uniform Noise Image');
```



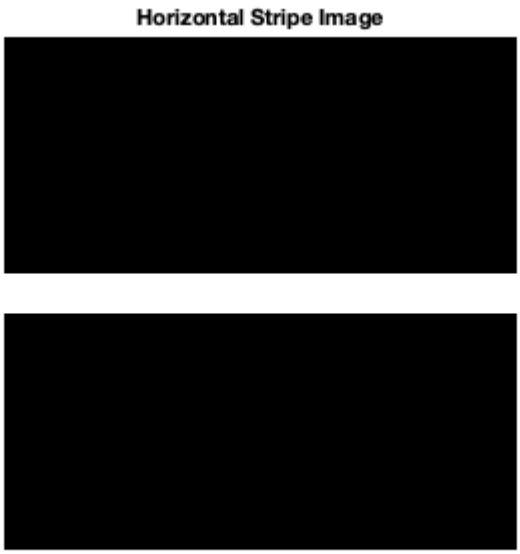

```
% Gaussin noise image
fg = testImage('gnoise',[M,N]);
figure; imshow(fg(1:2:M,1:2:N));
title('Gaussian Noise Image');
```




```
% Vertical stripe image
fv = testImage('vstripe',[M,N,90]);
figure; imshow(fv(1:2:M,1:2:N));
title('Vertical Stripe Image');
```




```
% Horizontal stripe image
fh = testImage('hstripe',[M,N,40]);
figure; imshow(fh(1:2:M,1:2:N));
title('Horizontal Stripe Image')
```



Problem 1.6: DIPUM3E Matlab Project 2.8

File info function using cell array

(a) `myfileinfoC` function:

```
function infoc = myimageinfoc(f)
% MYIMAGEINFOC  Numerical info mation about an image.
% INFOC = MYIMAGEINFOC(F) extraccs numerical information from image F
% and outputs it using the following cell array forms. In general, F
% can be of size M-by-N-by-P, with P >= 1 (for example, for RGB  images,
% p = 3).
%
% infoC{1} = number of rows in the image.
% infoc{2} = number of columns.
% infoC{3} = the number of image planes.
% infoc{4}(I) = maximum value of each image plane, I = 1:P.
% infoc{5}(I) = minimum value each image image, I = 1:P.

% Extract size and plane information info:
infoc{1} = size(f,1);
infoc{2} = size(f,2);
infoc{3} = size(f,3);

% calculate maximum and minimum values.
for I = 1:infoc{3}
    infoc{4}(I) = max(max(f(:,:,I)));
    infoc{5}(I) = min(min(f(:,:,I)));
end
end
```

(b) Application of `myimageinfoC` function

Read image **`ballerina.tif`**. Apply function **`myimageinfoC(f)`** to it and print out results.

Solution: MATLAB script:

```
clc; close all; clear;
f = imread('../artfiles/ballerina.tif');
figure; imshow(f(1:2:end,1:2:end,:));
title('Ballerina Image');
```


```
infoC = myimageinfoC(f); format short; disp(infoC)
```

```
    {[644]}    {[656]}    {[3]}    {1×3 uint8}    {1×3 uint8}
```

Problem 1.7

Let $f(m,n)$ be an input image, $h(m,n)$ be an impulse response, and $g(m,n)$ be the corresponding output image obtained via the 2-D convolution. Prove the following volume conservation property:

$$\sum_{m,n=-\infty}^{\infty}\sum g(m,n) = \left[\sum_{m,n=-\infty}^{\infty}\sum h(m,n)\right]\left[\sum_{m,n=-\infty}^{\infty}\sum f(m,n)\right]$$

Proof: Consider

$$
\begin{aligned}
\sum_{m,n=-\infty}^{\infty}\sum g(m,n) &= \sum_{m,n=-\infty}^{\infty}\sum \big[h(m,n) * f(m,n)\big] \\
&= \sum_{m,n=-\infty}^{\infty}\sum \left[\sum_{k,l=-\infty}^{\infty}\sum h(k,l)\, f(m-k,n-l)\right] \\
&= \sum_{k,l=-\infty}^{\infty}\sum \left[\sum_{m,n=-\infty}^{\infty}\sum h(k,l)\, f(m-k,n-l)\right] \\
&= \sum_{k,l=-\infty}^{\infty}\sum h(k,l)\left[\sum_{m,n=-\infty}^{\infty}\sum f(m-k,n-l)\right] \\
&= \left[\sum_{k,l=-\infty}^{\infty}\sum h(k,l)\right]\left[\sum_{m,n=-\infty}^{\infty}\sum f(m,n)\right] \\
&= \left[\sum_{m,n=-\infty}^{\infty}\sum h(m,n)\right]\left[\sum_{m,n=-\infty}^{\infty}\sum f(m,n)\right]
\end{aligned}
$$

Problem 1.8

Consider the following two-dimensional signal $f(m,n)$, where the bracketed element denotes the $(0,0)$ location

$$
f(m,n) = \begin{matrix} 8 & 5 & 3 & \\ 6 & 7 & 4 & \\ 4 & [7] & 6 & \rightarrow n \\ 3 & 5 & 8 & \\ & \downarrow & & \\ & m & & \end{matrix}
$$

(a) 2-D Convolution

Determine the convolution of $f(m,n)$ with each of the following signals

i. $h_1[m,n] = \begin{matrix} 0 & 1 & 0 \\ 1 & [-4] & 1 \\ 0 & 1 & 0 \end{matrix} \quad \Rightarrow \quad \sum\sum h_1(m,n) = 0.$

Solution: The 2-D convolution is

$$
g_1(m,n) = f_1[m,n] * h_1[m,n] = \begin{matrix} 0 & 8 & 5 & 3 & 0 \\ 8 & -21 & -2 & -3 & 3 \\ 6 & -5 & -6 & 0 & 4 \\ 4 & 0 & [-6] & -5 & 6 \\ 3 & -3 & -2 & -21 & 8 \\ 0 & 3 & 5 & 8 & 0 \end{matrix}
$$

ii. $h_2[m,n] = \begin{matrix} 1 & 0 & 0 \\ 0 & [2] & 0 \\ 0 & 0 & 1 \end{matrix} \Rightarrow \sum\sum h_2(m,n) = 6.$

Solution: The 2-D convolution is

$$g_2(m,n) = \begin{matrix} 8 & 5 & 3 & 0 & 0 \\ 6 & 23 & 14 & 6 & 0 \\ 4 & 19 & 28 & 13 & 3 \\ 3 & 13 & [28] & 19 & 4 \\ 0 & 6 & 14 & 23 & 6 \\ 0 & 0 & 3 & 5 & 8 \end{matrix}$$

iii. $h_3[m,n] = \begin{matrix} 0 & 2 \\ [-2] & 0 \\ 0 & 2 \end{matrix} \Rightarrow \sum\sum h_3(m,n) = 2.$

Solution: The 2-D convolution is

$$g_3(m,n) = \begin{matrix} 0 & 16 & 10 & 6 \\ -16 & 2 & 2 & 8 \\ -12 & 10 & 16 & 18 \\ -8 & [4] & 12 & 24 \\ -6 & -2 & -2 & 12 \\ 0 & 6 & 10 & 16 \end{matrix}$$

(b) Volume Conservation Property

Verify your answers in each case via the ***volume conservation*** property proved in **Problem 7** above.

Solution : Verification of volume conservation property:

$$\text{Vol}_f \triangleq \sum\sum f(m,n) = 66$$

i. $\text{Vol}_{h_1} \triangleq \sum\sum h_1(m,n) = 0$ and $\text{Vol}_{g_1} \triangleq \sum\sum g_1(m,n) = 0 = \text{Vol}_{h_1} \times \text{Vol}_f$.

ii. $\text{Vol}_{h_1} \triangleq \sum\sum h_2(m,n) = 4$ and $\text{Vol}_{g_2} \triangleq \sum\sum g_2(m,n) = 264 = \text{Vol}_{h_2} \times \text{Vol}_f$.

iii. $\text{Vol}_{h_3} \triangleq \sum\sum h_3(m,n) = 2$ and $\text{Vol}_{g_3} \triangleq \sum\sum g_3(m,n) = 132 = \text{Vol}_{h_3} \times \text{Vol}_f$.