Displacement of Object in Frequency Domain

By

Nidhi Gautam - 041011
Roli Saxena - 041054
Shivani Dubey -041043

May-2008

Submitted in partial fulfillment of the Degree of Bachelor of Technology

Department of Electronics And Communication Engineering

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY-WAKNAGHAT

CERTIFICATE

This is to certify that the work entitled, "Displacement of Object in Frequency Domain" submitted by Nidhi Gautam , Roli Saxena and Shivani Dubey on fulfillment for the award of degree of Bachelor Of Technology in Electronics and Telecommunication of Jaypee University Of Information Technology has been carried out under my supervision. This work has not been submitted partially or completely to any other university or institute for the award of this or any other degree or diploma.

Mr. Vinay Kumar

Prof. Sunil . V Bhooshan

ACKNOWLEDGEMENT

We would like to express our gratitude to all those who gave us the support and guidance to complete this project. We express our gratitude to Prof. V. Bhooshan HOD, Department of Electronics and Communication, JUIT for his help, support, interest and valuable hints during the project designing and building stage.

We are deeply indebted to our project guide Mr. Vinay Kumar Department of Electronics and Communication, whose help, stimulating suggestion and encouragement helped us in all the stages of the project. His overly enthusiasm and his view for providing 'only high quality work and not less', has made a deep impression on three of us.

Nidhi Gautam　　Roli Saxena　　Shivani Dubey

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This project describes a method for displacement or shifting of an object in an image in frequency domain. Here we have labeled the required object and then extracted it from the main image . Thereafter shifting of the extracted object is done using shifting property of the Fourier Transform.

INTRODUCTION

Brief Description

The Fourier Transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or frequency domain, while the input image is the spatial domain equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image.
The Fourier Transform is used in a wide range of applications, such as image analysis, image filtering, image reconstruction and image compression.

The One-Dimensional Fourier Transform and its Inverse

The Fourier transform, *F(u),* of a single variable, continuous function, *f(x),is* defined by the equation

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-j2\pi ux} dx \tag{1}$$

where $j = \sqrt{-1}$.
Conversely, given *F(u),* we can obtain *f(x)* by means of the inverse Fourier transform

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{j2\pi ux} du. \tag{2}$$

These two equations comprise the *Fourier transform pair.* They indicate the important fact that a function can be recovered from its transform.
These equations are easily extended to two variables, *u* and *v:*

$$F(u, v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(ux+vy)} dx\, dy. \tag{3}$$

and, similarly for the inverse transform,

$$f(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)} du\, dv. \tag{4}$$

The Fourier transform of a discrete function of one variable, $f(x)$, $x = 0, 1, 2, \ldots, M-1$, is given by the equation

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x) e^{-j2\pi ux/M} \qquad \text{for } u = 0, 1, 2, \ldots, M-1. \tag{5}$$

Similarly, given *F(u),* we can obtain the original function back, using the inverse DFT:

$$f(x) = \sum_{u=0}^{M-1} F(u)e^{j2\pi ux/M} \qquad \text{for } x = 0, 1, 2, \ldots, M-1. \tag{6}$$

The 1/ *M* multiplier in front of the Fourier transform sometimes is placed in front of the inverse instead. Other times (not as often) both equations are multiplied by I/√M. The location of the multiplier does not matter. If two multipliers are used, the only requirement is that their product be equal to 1/M. Considering their importance, these equations really are very simple.

In order to compute *F (u)* in Eq. (4.2-5) we start by substituting $u = 0$ in the exponential term and then summing for *all* values of *x*. We then substitute $u = 1$ in the exponential and repeat the summation over all values of *x*. We repeat this process for all *M* values of *u* in order to obtain the complete Fourier transform. It takes approximately M^2 summations and multiplications to compute the discrete Fourier transform. Like *f(x),* the transform is a discrete quantity, and it has the same number of components as *f(x).* Similar comments apply to the computation of the inverse Fourier transform.

An important property of the discrete transform pair is that, unlike the continuous case, we need not be concerned about the existence of the DFT or its in- verse. The discrete Fourier transform and its inverse always exist.

The concept of the frequency domain follows from Euler's formula:

$$e^{j\theta} = \cos\theta + j\sin\theta. \tag{7}$$

Substituting this expression into Eq. (5), and using the fact that,

cos(-Θ) = cosΘ ,

gives us:

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)[\cos 2\pi ux/M \; - \; j\sin 2\pi ux/M] \tag{8}$$

for $u = 0,1,2,\ldots, M-1$. Thus, we see that each term of the Fourier transform [that is, the value of *F(u)* for each value of *u]* is composed of the sum of *all* values of the function *f (x).* The values of *f (x),* in turn, are multiplied by sines and cosines of various frequencies. The domain (values of *u)* over which the values of F(u) range is appropriately called the frequency domain, because u determines the frequency of the components of the transform. (The x's also affect the in frequencies, but they are summed out and they all make the same contributions for each value of u.) Each of the M terms of F(u) is called a frequency component of the transform. Use of the terms frequency domain and frequency components is really no different from the terms time domain and time components, which we would use to express the domain and values of f(x) if x were a time variable.

The Two-Dimensional DFT and Its Inverse

Extension of the one-dimensional discrete Fourier transform and its inverse to two dimensions is straightforward. The discrete Fourier transform of a function (image) $f(x, y)$ of size M x N is given by the equation:

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}. \tag{9}$$

As in the 1-D case, this expression must be computed for values of u = 0,1,1 Z,...,M -1, and also for v = $0,1,2,...,N$-1.Similarly,givenF(u,v),weob..j tain $f(x, y)$ via the inverse Fourier transform, given by the expression:

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux/M + vy/N)} \tag{10}$$

for x = 0, 1,2,... , M-1 and y = 0, 1,2,... , N-1. Equations (9) and (10) comprise the two-dimensional, discrete Fourier transform (D FT) pair. The variables u and v are the transform or frequency variables, and x and yare the spatial or image variables. As in the one-dimensional case, the location of the 1/ M N constant is not important. Sometimes it is located in front of the inverse transform. Other times it is found split into two equal terms of $I/\sqrt{MN}$ multiplying the transform and its inverse.

The definition of the Fourier spectrum, phase angle, and power spectrum are as under:

$$|F(u, v)| = \left[R^2(u, v) + I^2(u, v)\right]^{1/2} \tag{11}$$

$$\phi(u, v) = \tan^{-1}\left[\frac{I(u, v)}{R(u, v)}\right] \tag{12}$$

$$\begin{aligned} P(u, v) &= |F(u, v)|^2 \\ &= R^2(u, v) + I^2(u, v) \end{aligned} \tag{13}$$

where $R(u, v)$ and $l(u, v)$ are the real and imaginary parts of $F(u, v)$, respectively.

It is common practice to multiply the input image function by (-1)^(x+y) prior to computing the Fourier transform. Due to the properties of exponentials, it is shown that:

$$\Im\left[f(x, y)(-1)^{x+y}\right] = F(u - M/2, v - N/2) \tag{14}$$

where ℑ[.] denotes the Fourier transform of the argument. This equation states that the origin of the Fourier transform of f(x, y)(-1)^(x+y) [that is,F(0, 0)] is located at u = M/2 and v = N /2. In other words, multiplying f(x, y) (-1)^(x+y) by shifts the origin of F(u, v) to frequency coordinates (M/2, N /2), which is the center of the M x N area occupied by the 2-D DFT. We refer to this area of the frequency domain as the frequency rectangle extends from u = 0 to u = M -1, and from v = 0 to v = N -1 (u and v are integers).
In order to guarantee that these shifted coordinates are integers, we require that M and N be even numbers. When implementing the Fourier transform in a computer, the limits of summations are from u = 1 to M and v = 1 to N. The actual center of the transform will then be at u= (M/2) + 1 and v = (N/2) + 1.

The value of the transform at *(u, v)* = (0,0) is, from Eq. (9):

$$F(0,0) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y), \qquad (15)$$

which is the average of f(x, y). In other words, if f(x, y) is an image, the value of the Fourier transform at the origin is equal to the average gray level of the image. Because both frequencies are zero at the origin, F(0, 0) sometimes is called the dc component of the spectrum. This terminology is from electrical engineering, where "dc" signifies direct current (i.e., current of zero frequency).

If *f(x, y)* is real, its Fourier transform is conjugate symmetric; that is,

$$F(u, v) = F^*(-u,-v) \qquad (16)$$

where "*" indicates the standard conjugate operation on a complex number.

From this, it follows that

$$|F(u, v)| = |F(-u,-v)| \qquad (17)$$

which says that the spectrum of the Fourier transform is symmetric.

We have the following relationships between samples in the spatial and frequency domains:

$$\Delta u = 1/(M \Delta x) \quad (18)$$

And

$$\Delta v = 1/(N \Delta y) \quad (19)$$

Properties of the 2-D Fourier Transform

-Translation

The Fourier transform pair has the following translation properties:

$$f(x, y)e^{j2\pi(u_0x/M + v_0y/N)} \Leftrightarrow F(u - u_0, v - v_0) \quad (20)$$

and

$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M + vy_0/N)} \quad (21)$$

where, the double arrow is used to designate a Fourier transform pair.
When *Uo* = M /2 and *Vo* = *N/2,* it follows that :

$$e^{j2\pi(u_0x/M + v_0y/N)} = e^{j\pi(x+y)}$$
$$= (-1)^{x+y} \quad (22)$$

In this case, Eq. (20) becomes

$$f(x, y)(-1)^{x+y} \Leftrightarrow F(u - M/2, v - N/2) \quad (23)$$

and, similarly,

$$f(x - M/2, y - N/2) \Leftrightarrow F(u, v)(-1)^{(u+v)} \quad (24)$$

We see that Eq. (23) is the same as Eq. (14), which we used for centering the transform. These results are based on the variables u and v having values in the range [0, M -1] and [0, N -1], respectively. In a computer implementation these variables will run from u = 1 to M and v = 1 to N, in which case the actual center of the transform will be at
u = (M /2) + 1 and v = (N /2) + 1.

-Distributivity and scaling

From the definition of the Fourier transform it follows that

$$\Im[f_1(x, y) + f_2(x, y)] = \Im[f_1(x, y)] + \Im[f_2(x, y)] \quad (25)$$

and, in general, that

$$\Im[f_1(x,y)\cdot f_2(x,y)] \neq \Im[f_1(x,y)]\cdot\Im[f_2(x,y)]. \quad (26)$$

In other words, the Fourier transform is distributive over addition, but not over multiplication. Identical comments apply to the inverse Fourier transform. Similarly, for two scalars a and b,

$af(x, y) \Leftrightarrow aF(u, v)$ *(27)*

and

$$f(ax, by) \Leftrightarrow \frac{1}{|ab|}F(u/a, v/b) \quad (28)$$

-Rotation

If we introduce the polar coordinates

$x = r\cos\Theta$ $y = r\sin\Theta$ $u = \omega\cos\psi$ $v = \omega\sin\psi$
then $f(x, y)$ and $F(u, v)$ become $f(r,\Theta)$ and $F(\omega, \psi)$, respectively.
Direct substitution into the definition of the Fourier transforms yields

$$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0) \quad (29)$$

This expression indicates that rotating $f(x, y)$ by an angle Θo rotates $F(u, v)$ by the same angle. Similarly, rotating $F(u, v)$ rotates $f(x, y)$ by the same angle.

-Periodicity and Conjugate symmetry

The discrete Fourier transform has the following periodicity properties:
$F(u, v) = F(u + M, v) = F(u, v + N) = F(u + M, v + N)$. (30)

The inverse transform is also periodic:
$f(x, y) = f(x + M, y) = f(x, y + N) = f(x + M, y + N)$. (31)

-Conjugate symmetry is given as :

$F(u, v) = F^*(-u, -v)$ (32)
From which it follows that the spectrum also is symmetric about the origin:
$|F(u,v)| = |F(-u,-v)|$. (33)
The validity of these equations is easily established from Eqs. (9) and (10).

Digital Image Definitions

A digital image a[m,n] described in a 2D discrete space is derived from an analog image a(x,y) in a 2D continuous space through a sampling process that is frequently referred to as digitization.

The 2D continuous image a(x,y) is divided into N rows and M columns. The intersection of a row and a column is termed a pixel. The value assigned to the integer coordinates [m,n] with {m=0,1,2,...,M-1} and {n=0,1,2,...,N-1} is a[m,n]. In fact, in most cases a(x,y)--which we might consider to be the physical signal

that impinges on the face of a 2D sensor--is actually a function of many variables including depth (z), color (λ), and time (t).

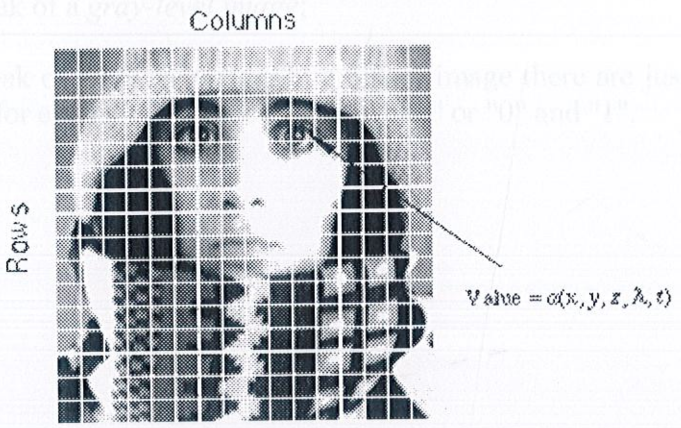


Figure-1- Image explaining pixel value

The image shown in Figure 1 has been divided into N = 16 rows and M = 16 columns. The value assigned to every pixel is the average brightness in the pixel rounded to the nearest integer value. The process of representing the amplitude of the 2D signal at a given coordinate as an integer value with L different gray levels is usually referred to as amplitude quantization or simply quantization.
There are standard values for the various parameters encountered in digital image processing. These values can be caused by video standards, by algorithmic requirements, or by the desire to keep digital circuitry simple.

Table 1 gives some commonly encountered values.

Parameter	*Symbol*	*Typical values*
Rows	*N*	256,512,525,625,1024,1035
Columns	*M*	256,512,768,1024,1320
Gray Levels	*L*	2,64,256,1024,4096,16384

Table 1: Common values of digital image parameters

The number of distinct gray levels is usually a power of 2, that is, $L=2^B$ where B is the number of bits in the binary representation of the brightness levels.

When $B>1$ we speak of a *gray-level image*;

When $B=1$ we speak of a *binary image*. In a binary image there are just two gray levels which can be referred to, for example, as "black" and "white" or "0" and "1".

PROCEDURE

We have developed an approach to shift the objects in the image using shifting property of Fourier transform.

$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M + vy_0/N)} \quad (1)$$

The procedure includes the following steps:

Step 1:

```
M=imread('final.png');
I = rgb2gray(M);
```

The imread command returns the image (final.png) data in the array M.
Thereafter rgb2gray converts rgb image to gray scale image whose data is stored in array I.

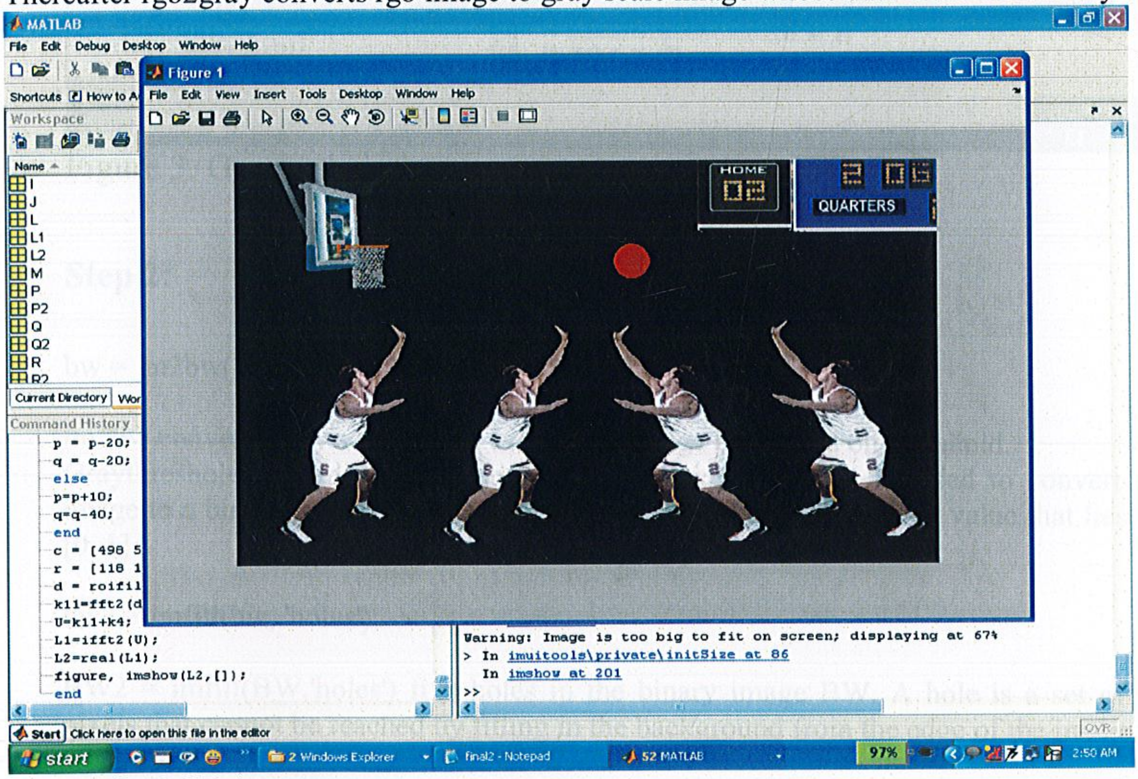


Figure 2. RGB image

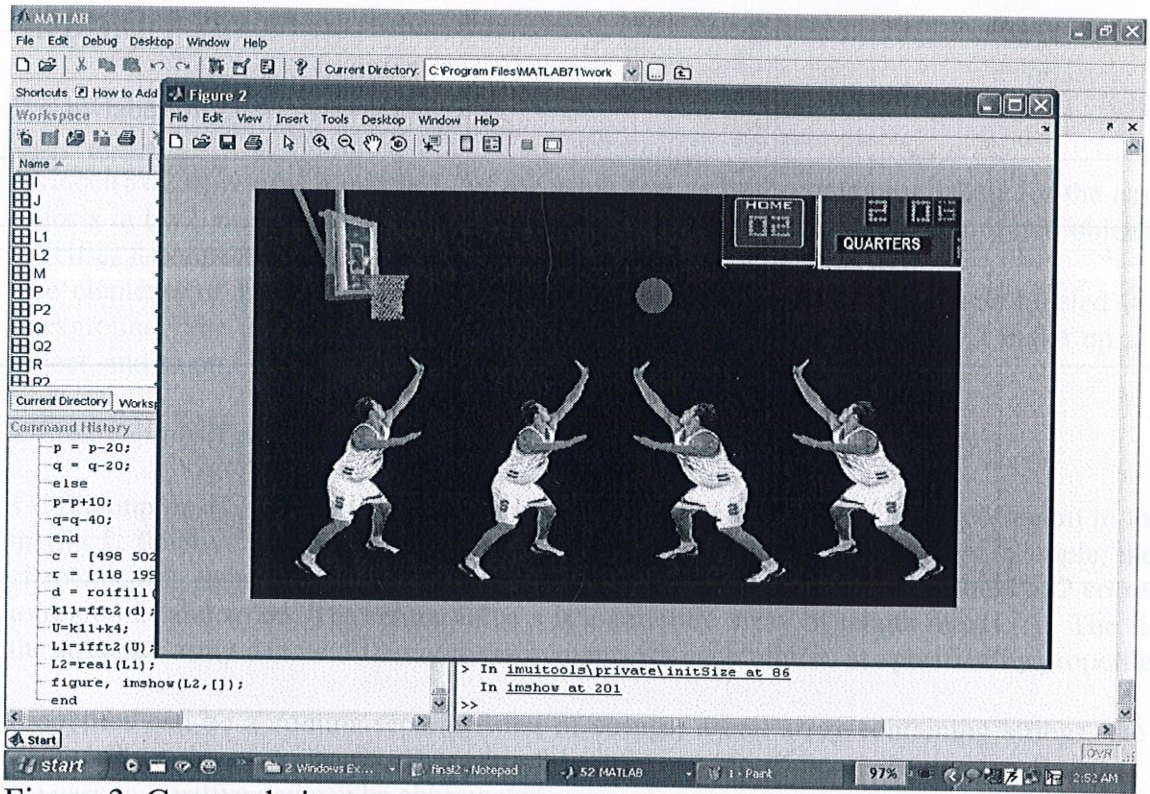


Figure 3. Grayscale image

Step 2:

bw = im2bw(I, graythresh(I));

im2bw converts gray scale image I to binary image bw, based on threshold.
Graythreshold(I) computes a global threshold(level) that can be used to convert an intensity image to a binary image with im2bw. Level is a normalized intensity value that lies in the range [0, 1].

bw2 = imfill(bw, 'holes');

BW2 = imfill(BW,'holes') fills holes in the binary image BW. A hole is a set of background pixels that cannot be reached by filling in the background from the edge of the image.

Step 3:

L = bwlabel(bw2);

bwlabel(bw2,n) returns a matrix L, of the same size as bw2, containing labels for the connected objects in bw2. n can have a value of either 4 or 8, where 4 specifies 4-connected objects and 8 specifies 8-connected objects; if the argument is omitted, it defaults to 8.
The elements of L are integer values greater than or equal to 0. The pixels labeled 0 are the background. The pixels labeled 1 make up one object, the pixels labeled 2 make up a second object, and so on.

s = regionprops(L, 'Centroid');

s = regionprops(L, 'properties') measures a set of properties for each labeled region in the label matrix L. Positive integer elements of L correspond to different regions. For example, the set of elements of L equal to 1 corresponds to region 1; the set of elements of L equal to 2 corresponds to region 2; and so on. The return value s is a structure array of length max(L(:)). The fields of the structure array denote different measurements for each region, as specified by properties.

Properties can be a comma-separated list of strings, a cell array containing strings, the single string 'all', or the string 'basic'. This table-2 lists the set of valid property strings. Property strings are case insensitive and can be abbreviated.

'Area'	'EulerNumber'	'Orientation'
'BoundingBox'	'Extent'	'Perimeter'
'Centroid'	'Extrema'	'PixelIdxList'
'ConvexArea'	'FilledArea'	'PixelList'
'ConvexHull'	'FilledImage'	'Solidity'
'ConvexImage'	'Image'	'SubarrayIdx'
'Eccentricity'	'MajorAxisLength'	
'EquivDiameter'	'MinorAxisLength'	

Table – 2 – regionprops properties

'Centroid'-- 1-by-ndims(L) vector; the center of mass of the region. The first element of Centroid is the horizontal coordinate (or x-coordinate) of the center of mass, and the second element is the vertical coordinate (or y-coordinate). All other elements of Centroid are in order of dimension.
For our progam each element of the array s is a structure with one field that is centroid. Size of the array s is 1x 150.

Step 4:

```
figure,imshow(bw2)

hold on
for k = 1:numel(s)
c = s(k).Centroid;
text(c(1), c(2), sprintf('%d', k), ...
'HorizontalAlignment', 'center', ...
'VerticalAlignment', 'middle');
end
hold off
```

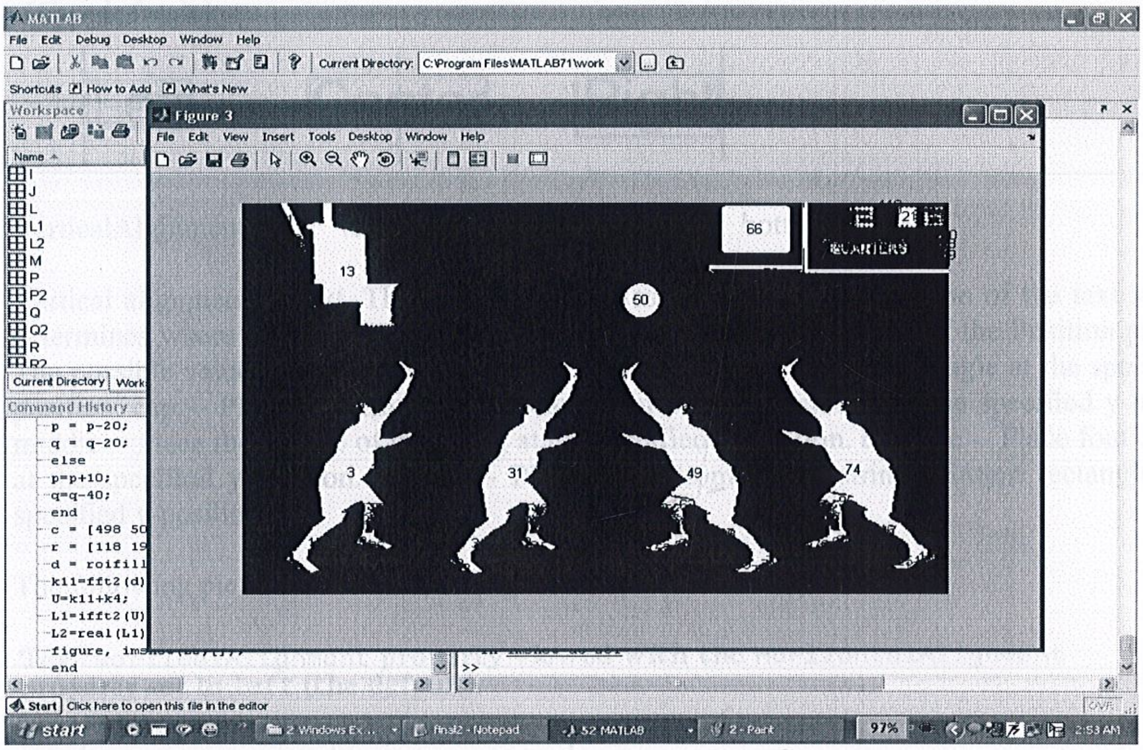


Figure 4. Labelled image

hold on retains the current image bw2 and certain axes properties so that subsequent graphing commands add to the existing image.

Numel is Number of elements in array or subscripted array expression.

text(x, y, z, 'string', 'PropertyName', PropertyValue ....) adds the string - sprintf('%d', k) in quotes to the location defined by the coordinates c(1), c(2) and uses the values for the specified text properties.

Here we are using Horizontal Alignment and VerticalAlignment

HorizontalAlignment {left} | center | right

Horizontal alignment of text. This property specifies the horizontal justification of the text string. It determines where MATLAB places the string with regard to the point specified by the Position property.

The following picture illustrates the alignment options.

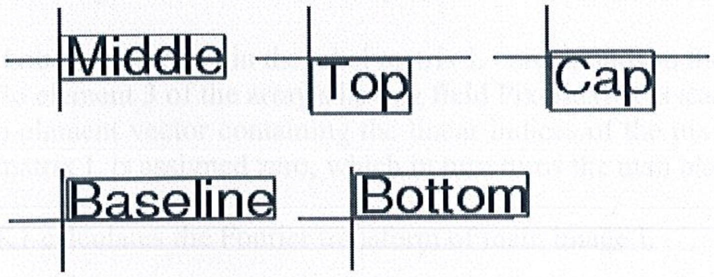
HorizontalAlignment viewed with the VerticalAlignment set to middle (the default).

VerticalAlignment top | cap | {middle} | baseline | bottom

Vertical alignment of text. This property specifies the vertical justification of the text string. It determines where MATLAB places the string with regard to the value of the Position property. The possible values mean top -- Place the top of the string's Extent rectangle at the specified y-position. cap -- Place the string so that the top of a capital letter is at the specified y-position. middle -- Place the middle of the string at the specified y-position. baseline -- Place font baseline at the specified y-position. bottom -- Place the bottom of the string's Extent rectangle at the specified y-position…

The following picture illustrates the alignment options.

Text VerticalAlignment property viewed with the HorizontalAlignment property set to left (the default).



Step 5:

s = regionprops(L, 'PixelIdxList')

'PixelIdxList' -- p-element vector containing the linear indices of the pixels in the for each labeled region in the label matrix L.

Now each element of the array s is a structure with one field that is pixelidxlist.
Size of the array s is 1x 150.

Step 6:

```
%For the 1st man
J = I;
%To Turn the 1st man black
J(s(3).PixelIdxList) = 0;
figure,imshow(J);

% fourier transform of main image
k1=fft2(I);

% fourier transform the image with 1st man blackened
k2=fft2(J);

% fourier transform of image with the 1st man only
k3=k1-k2;
y3=ifft2(k3);
y4=real(y3);
figure, imshow(y4,[]);
```

We assign the main image array I to J.

Labeled region 3 in the label matrix L corresponds to the 1st man.
So element 3 of the array s having field PixelIdxlist is assigned zero value .
p-element vector containing the linear indices of the pixels in the labeled region 3 in the label matrix L is assigned zero, which in turn turns the man black.

K1 calculates the Fourier transform of main image I.

K2 calculates Fourier transform the image with 1st man blackened J.

K3 that is Fourier transform of image with the 1st man only is obtained by subtracting K2 from K1.
We then calculate inverse Fourier transform of k3 which is stored in the matrix y3 and then the real part of y3 stored in matrix y4.

Using imshow(y4,[]) command we display the image with 1st man only.

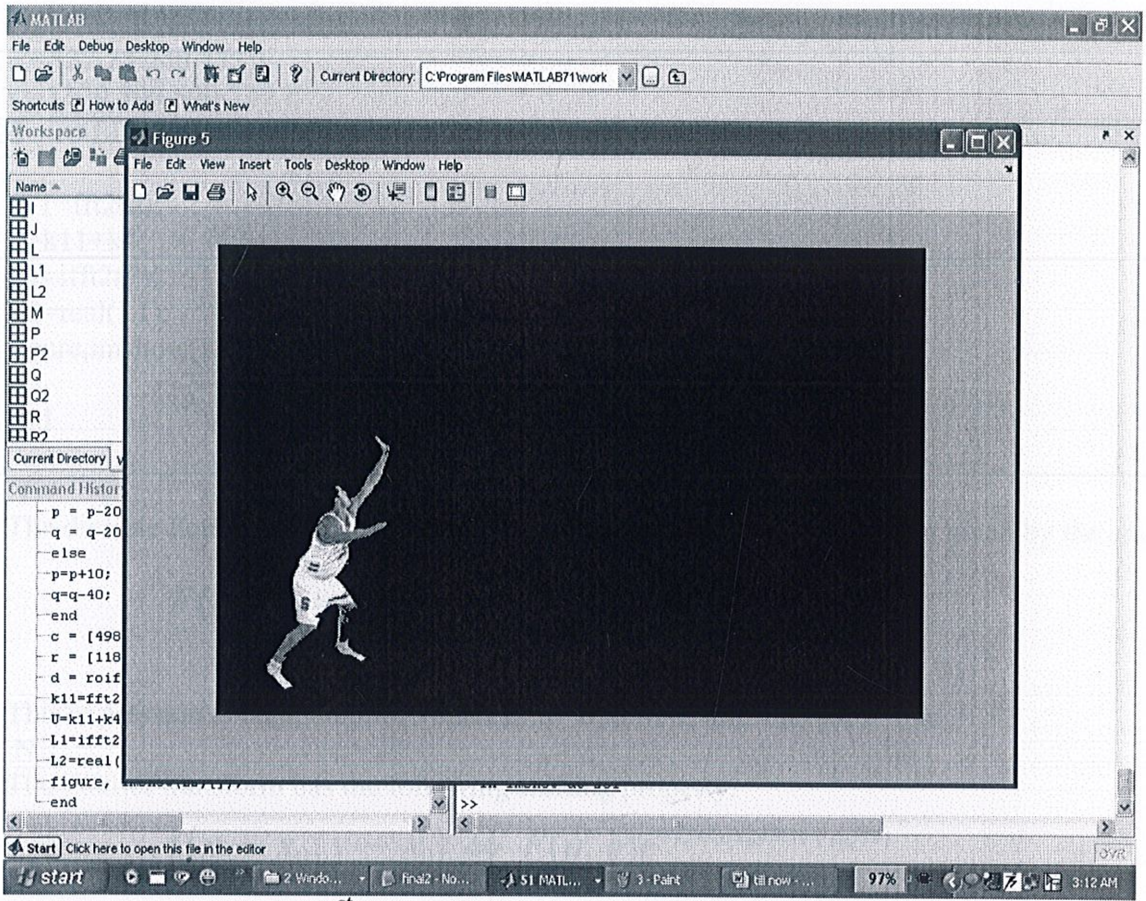


Figure 5. Image with 1st man only

Step 7:

```
% Loop to display movements of 1st man
Counter1=1;
Counter2=1;
for a = 1:10:50

    for u = 1: 610
    for v = 1:958

  k4 (u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 +(v*counter2)/958)));

    end
    end

f1=ifft2(k4);
```

```
f2=real(f1);

counter2 = counter2+10;

c = [32 23 240 272];
r = [229 590 590 227 ];
d = roifill(I,c,r);

k11=fft2(d);
P=k11+k4;
L1=ifft2(P);
L2=real(L1);
figure,imshow(L2,[]);
end
```

The discrete Fourier transform of a function (image) f(x,y) of size MxN is given by the equation:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}$$

This expression is computed for values of u= 0 to M-1, and also for v=0 to N-1.

The Fourier transform has the following shifting property:

$$f(x-x_0,y-y_0) \Leftrightarrow F(u,v)e^{-j2\pi(ux_0/M+vy_0/N)}$$

where xo = counter1 and yo= counter2.

F(u,v)= k3(u,v) that is Fourier transform of image with the 1st man only

M= 610 , N= 958.

Here two variables are taken (counter1, counter2) to control the movement of selected object i.e. 1st man . Initially their values are 1 which implies the initial position of the object which are incremented/decremented according to the desired direction of selected object. In this particular case counter 1 is kept constant and counter2 is incremented with a value 10. This results in the movement in the positive x direction.

Then another variable a is used to control the number of movements of the selected object. In the case of first man the number of movements are 5, so a is assigned the value 1 to 50 with step increment of 10.

Within first for loop, we apply the shifting property of the Fourier transform using u and v for loops ,The values of the u and v loop are assigned according to the size of the image taken which is 610 x 958.

```
for u = 1: 610
   for v = 1:958

  k4 (u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 +(v*counter2)/958)));

   end
   end
```

k4(u,v) gives the fourier transform of the image with the shifted object according to the value of counter1 and counter2.

We then calculate inverse Fourier transform of k4 which is stored in the matrix f1 and then the real part of f1 which is stored in f2.
Now the roifill(I,c,r) command fills in the polygon specified by c and r, which are equal-length vectors containing the row-column coordinates of the pixels on vertices of the polygon.
Here c and r, contain the row and column coordinates of the polygon containing 1st man.

```
c = [32 23 240 272];
r =[229 590 590 227 ];
d = roifill(I,c,r);
```

Now we calculate the Fourier transform k11 of the image d ,obtained after applying roifill command .
Then we add the k11 and k4 which is stored in the matrix P.

We then calculate inverse Fourier transform of P which is stored in the matrix L1 and then the real part of P stored in matrix L2.

Using imshow(L2,[]) command we display the image with shifted object.
Now according to the value of a, the a loop executes 5 times which gives 5 movements of the selected object .

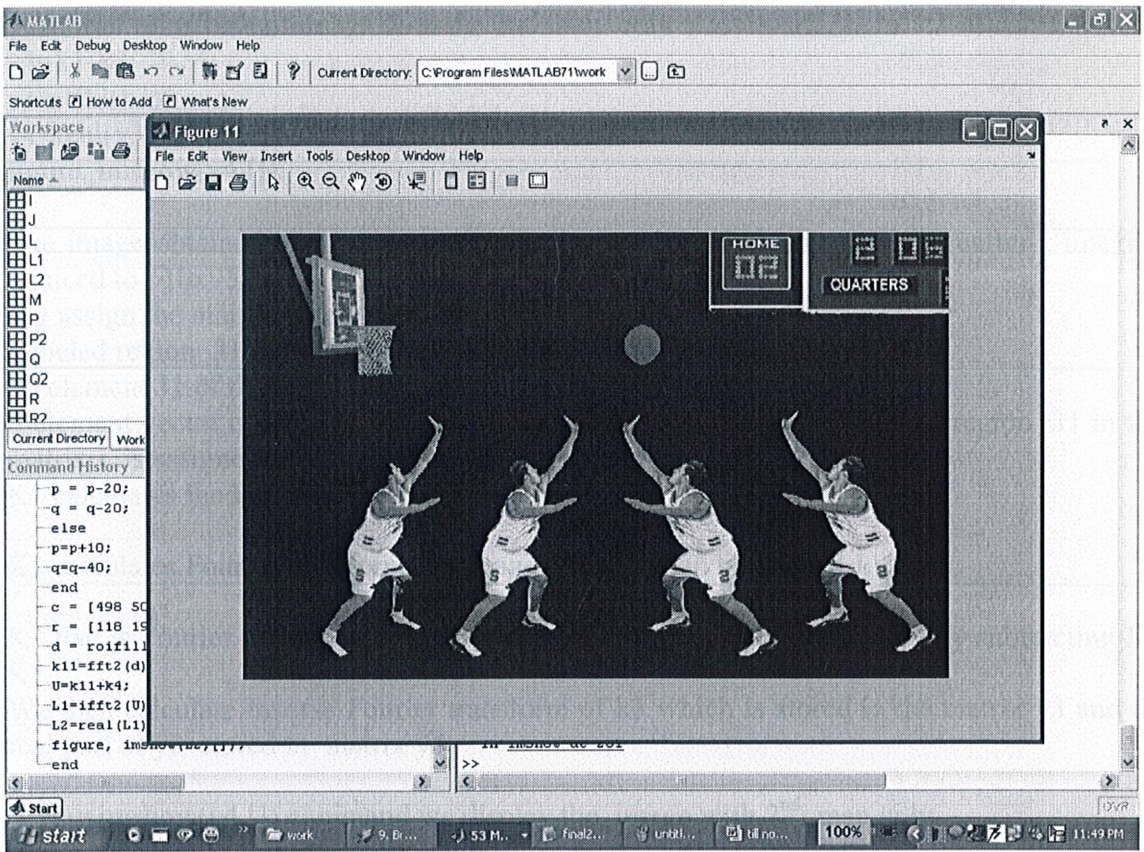


Figure 6. Image with 1st man shifted.

Step 8:


```
%For the 2nd man
P2=imread('shift1.png');
P2=P2(34:643,92:1049);
figure,imshow(P2);

J = I;
% Turn the 2nd man black
J(s(31).PixelIdxList) = 0;
figure,imshow(J);

% fourier transform of main image
k1=fft2(I);

% fourier transform the image with 2nd man blackened
k2=fft2(J);
```

```
% fourier transform of image with the 2nd man only
k3=k1-k2;
y3=ifft2(k3);
y4=real(y3);
figure, imshow(y4,[]);
```

The image obtained after shifting 1st man is saved as shift1.png and thereafter image size is reduced to 610x958.
We assign the main image array I to J.
Labeled region 31 in the label matrix L corresponds to the 2nd man.
So element 31 of the array s having field PixelIdxlist is assigned zero value .
p-element vector containing the linear indices of the pixels in the labeled region 31 in the label matrix L is assigned zero, which in turn turns the man black.
K1 calculates the Fourier transform of main image I.

K2 calculates Fourier transform the image with 1st man blackened J.

K3 that is Fourier transform of image with the 2nd man only is obtained by subtracting K2 from K1.
We then calculate inverse Fourier transform of k3 which is stored in the matrix y3 and then the real part of y3 stored in matrix y4.

Using imshow(y4,[]) command we display the image with 2nd man only.

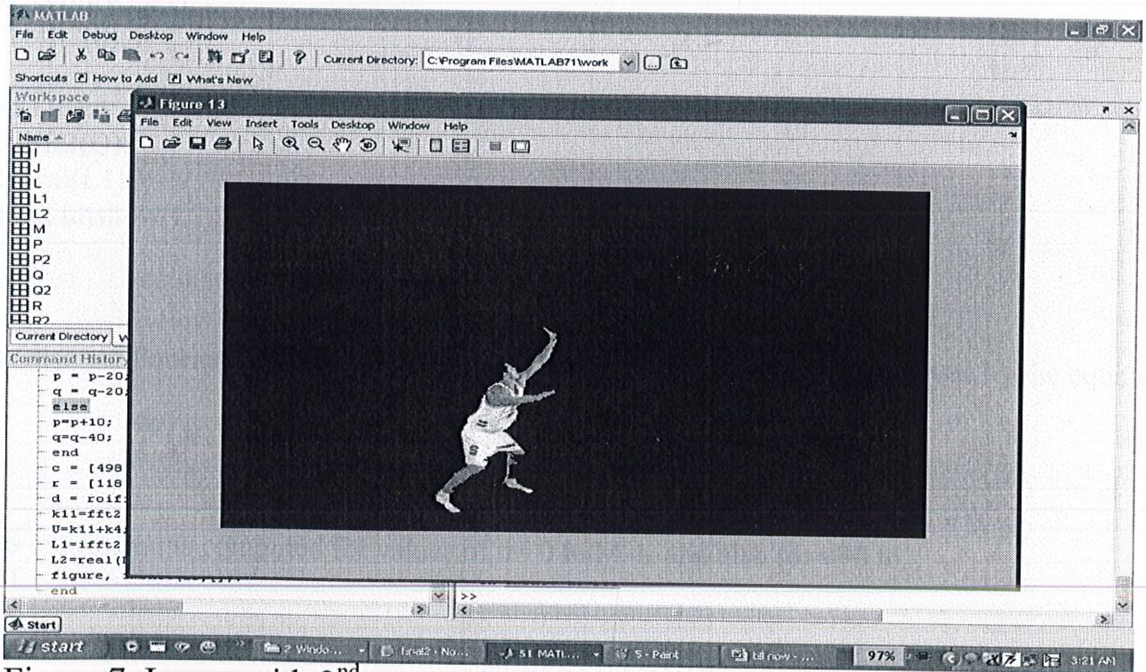


Figure 7. Image with 2nd man only.

Step 9:

```
% Loop to display movements of 2nd man
Counter1=1;
Counter2=1;

for a = 1:10:50

    for u = 1: 610
    for v = 1:958

  k4 (u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610+(v*counter2)/958)));

    end
    end

 f1=ifft2(k4);
 f2=real(f1);

 counter1 = counter1-30;

 c = [306 293 248 482 464];
 r = [234 483 587 589 221];
 d = roifill(P2,c,r);
 k11=fft2(d);


 Q=k11+k4;
 L1=ifft2(Q);
 L2=real(L1);
 figure, imshow(L2,[]);

end
```

The discrete Fourier transform of a function (image) f(x,y) of size MxN is given by the equation:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}.$$

This expression is computed for values of u= 0 to M-1, and also for v=0 to N-1.

The Fourier transform has the following shifting property:

$$f(x-x_0, y-y_0) \Leftrightarrow F(u,v)e^{-j2\pi(ux_0/M+vy_0/N)}.$$

where xo = counter1 and yo= counter2.

$F(u,v) = k3(u,v)$ that is Fourier transform of image with the 2nd man only
M= 610 , N= 958.

Here two variables are taken (counter1, counter2) to control the movement of selected object i.e. 2nd man . Initially their values are 1 which implies the initial position of the object which are incremented/decremented according to the desired direction of selected object. In this particular case counter 2 is kept constant and counter1 is decremented with a value -30.This results in the movement in the positive y direction.

Then another variable a is used to control the number of movements of the selected object. In the case of first man the number of movements are 5, so a is assigned the value 1 to 50 with step increment of 10.

Within first for loop, we apply the shifting property of the Fourier transform using u and v for loops .The values of the u and v loop are assigned according to the size of the image taken which is 610 x 958.

```
for u = 1: 610
    for v = 1:958

  k4 (u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 +(v*counter2)/958)));

    end
    end
```

k4(u,v) gives the Fourier transform of the image with the shifted object according to the value of counter1 and counter2.

We then calculate inverse Fourier transform of k4 which is stored in the matrix f1 and then the real part of f1 which is stored in f2.
Now the roifill(P2,c,r) command fills in the polygon specified by c and r, which are equal-length vectors containing the row-column coordinates of the pixels on vertices of the polygon.
Here c and r, contain the row and column coordinates of the polygon containing 2nd man.

```
c = [306 293 248 482 464];
r = [234 483 587 589 221];
d = roifill(P2,c,r);
```

Now we calculate the fourier transform k11 of the image d ,obtained after applying roifill command .
then we add the k11 and k4 which is stored in the matrix Q.

We then calculate inverse Fourier transform of Q which is stored in the matrix L1 and then the real part of P stored in matrix L2.

Using imshow(L2,[]) command we display the image with shifted object.
Now according to the value of a , the a loop executes 5 times which gives 5 movements of the selected object .

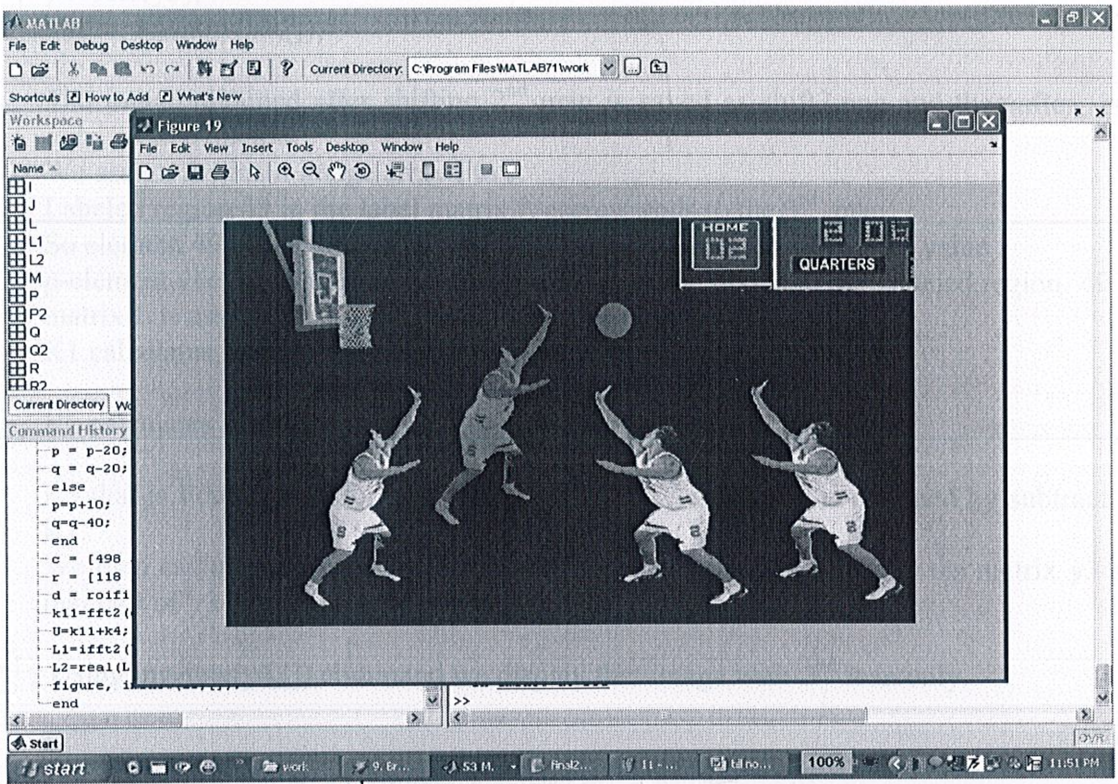


Figure 8. Image with 2nd man shifted.

Step 10:

```
%For the 3rd man
Q2=imread('shift2.png');
Q2=Q2(34:643,92:1049);
figure,imshow(Q2);

J = I;
% Turn the 3rd man black
J(s(49).PixelIdxList) = 0;
figure,imshow(J);

% fourier transform of main image
k1=fft2(I);

% fourier transform the image with 3rd man blackened
k2=fft2(J);
```

```
% fourier transform of image with the 3rd man only
k3=k1-k2;
y3=ifft2(k3);
y4=real(y3);
figure, imshow(y4,[]);
```

The image obtained after shifting 2nd man is saved as shift2.png and thereafter image size is reduced to 610x958.
We assign the main image array I to J.
Labeled region 49 in the label matrix L corresponds to the 3rd man.
So element 49 of the array s having field PixelIdxlist is assigned zero value .
p-element vector containing the linear indices of the pixels in the labeled region 49 in the label matrix L is assigned zero, which in turn turns the man black.
K1 calculates the Fourier transform of main image I.

K2 calculates Fourier transform the image with 1st man blackened J.

K3 that is Fourier transform of image with the 3rd man only is obtained by subtracting K2 from K1.
We then calculate inverse Fourier transform of k3 which is stored in the matrix y3 and then the real part of y3 stored in matrix y4.

Using imshow(y4,[]) command we display the image with 3rd man only.

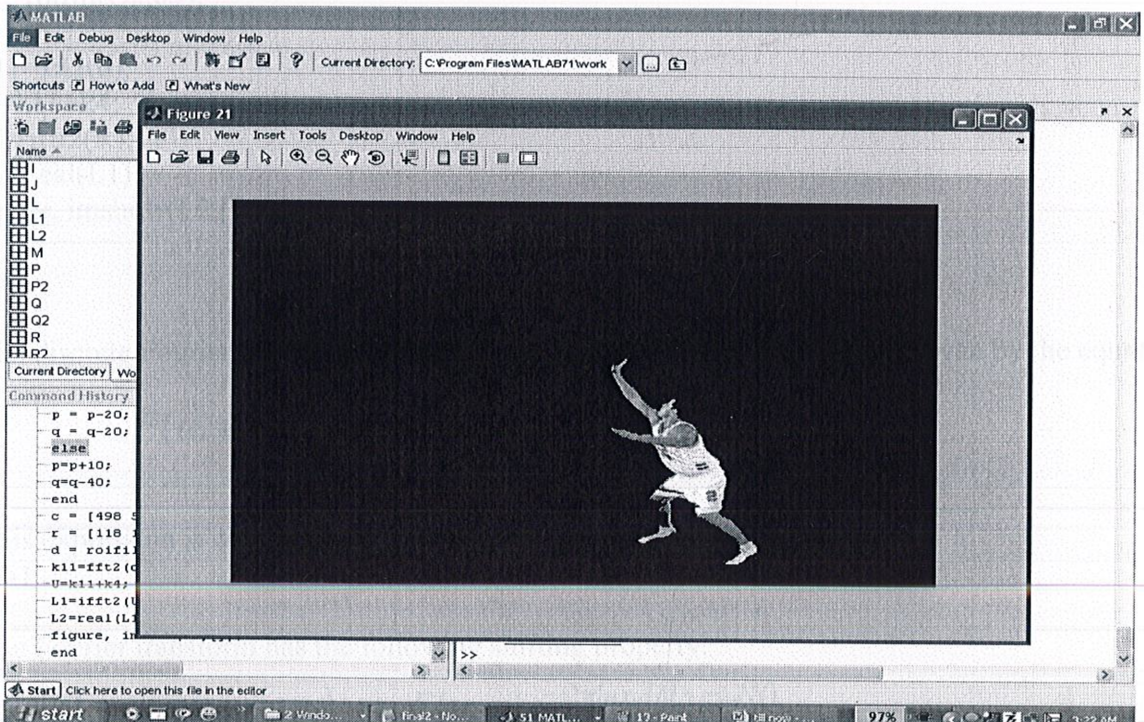


Figure 9. Image with 3rd man only.

Step 11:

```
% Loop to display movements of 3rd man
Counter1=1;
Counter2=1;

for a = 1:10:50

    for u = 1: 610
    for v = 1:958
 k4(u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 + (v*counter2)/958 )));

    end
    end

f1=ifft2(k4);
f2=real(f1);

counter2 = counter2-10;

c = [484 508 743 666];
r = [243 582 584 224];
d = roifill(Q2,c,r);

k11=fft2(d);
R=k11+k4;
L1=ifft2(R);
L2=real(L1);
figure, imshow(L2,[]);

end
```

The discrete Fourier transform of a function (image) f(x,y) of size MxN is given by the equation:

$$F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M + vy/N)}.$$

This expression is computed for values of u= 0 to M-1, and also for v=0 to N-1 .

The Fourier transform has the following shifting property:

$$f(x - x_0, y - y_0) \Leftrightarrow F(u,v) e^{-j2\pi(ux_0/M + vy_0/N)}.$$

where xo = counter1 and yo= counter2.

F(u,v)= k3(u,v) that is Fourier transform of image with the 1st man only
M= 610 , N= 958.

Here two variables are taken (counter1, counter2) to control the movement of selected object i.e. 3rd man . Initially their values are 1 which implies the initial position of the object which are incremented/decremented according to the desired direction of selected object. In this particular case counter 1 is kept constant and counter2 is decremented with a value -10.this results in the movement in the negative x direction.

Then another variable a is used to control the number of movements of the selected object. In the case of third man the number of movements are 5, so a is assigned the value 1 to 50 with step increment of 10.

Within first for loop, we apply the shifting property of the Fourier transform using u and v for loops .The values of the u and v loop are assigned according to the size of the image taken which is 610 x 958.

```
for u = 1: 610
    for v = 1:958

  k4 (u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 +(v*counter2)/958)));

    end
    end
```

k4(u,v) gives the fourier transform of the image with the shifted object according to the value of counter1 and counter2.

We then calculate inverse Fourier transform of k4 which is stored in the matrix f1 and then the real part of f1 which is stored in f2.

Now the roifill(Q2,c,r) command fills in the polygon specified by c and r, which are equal-length vectors containing the row-column coordinates of the pixels on vertices of the polygon.
Here c and r, contain the row and column coordinates of the polygon containing 3rd man.
c = [484 508 743 666];
r = [243 582 584 224];
d = roifill(Q2,c,r);
Now we calculate the Fourier transform k11 of the image d ,obtained after applying roifill command .

Then we add the k11 and k4 which is stored in the matrix R.
We then calculate inverse Fourier transform of R which is stored in the matrix L1 and then the real part of P stored in matrix L2.

Using imshow(L2,[]) command we display the image with shifted object.
Now according to the value of a , the a loop executes 5 times which gives 5 movements of the selected object .

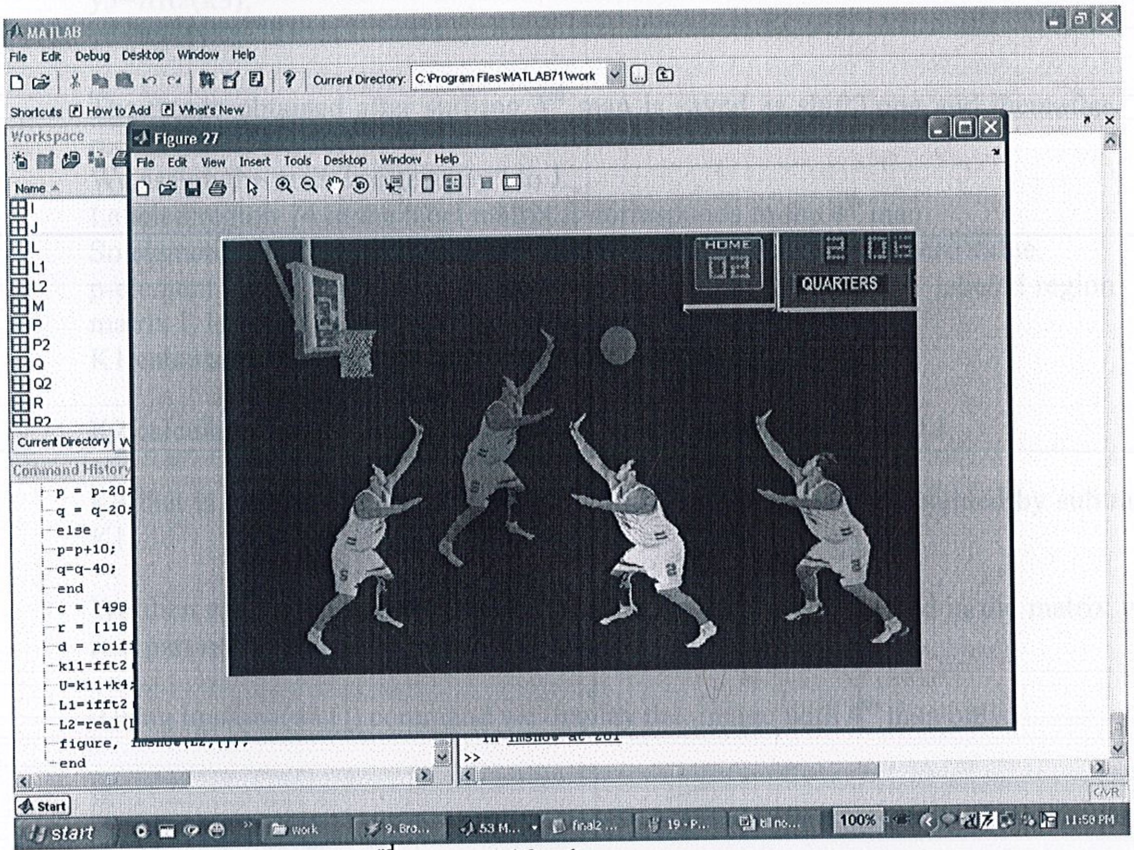


Figure 10. Image with 3[rd] man shifted.

Step 12:

```
%For the 4th man
R2=imread('shift3.png');
R2=R2(34:643,92:1049);
figure,imshow(R2);

J = I;
% Turn the 4th man black
J(s(74).PixelIdxList) = 0;
figure,imshow(J);

% fourier transform of main image
k1=fft2(I);

% fourier transform the image with 4th man blackened
k2=fft2(J);
```

% fourier transform of image with the 4th man only
k3=k1-k2;
y3=ifft2(k3);
y4=real(y3);

The image obtained after shifting 3rd man is saved as shift3.png and thereafter image size is reduced to 610x958.
We assign the main image array I to J.
Labeled region 74 in the label matrix L corresponds to the 4th man.
So element 74 of the array s having field PixelIdxlist is assigned zero value.
p-element vector containing the linear indices of the pixels in the labeled region 74 in the label matrix L is assigned zero, which in turn turns the man black.
K1 calculates the Fourier transform of main image I.

K2 calculates Fourier transform the image with 1st man blackened J.

K3 that is Fourier transform of image with the 4th man only is obtained by subtracting K2 from K1.

We then calculate inverse Fourier transform of k3 which is stored in the matrix y3 and then the real part of y3 stored in matrix y4.

Using imshow(y4,[]) command we display the image with 4th man only.

Figure 11. Image with 4th man only.

Step 13:

```
% Loop to display movements of 4th man
Counter1=1;
Counter2=1;

for a = 1:10:90

    for u = 1: 610
    for v = 1:958

 k4(u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 + (v*counter2)/958 )));

    end
    end

f1=ifft2(k4);
f2=real(f1);

counter1 = counter1-10;
counter2= counter2-20;

c = [682 754 950 935];
r = [227 569 586 210];
d = roifill(R2,c,r);

k11=fft2(d);
T=k11+k4;
L1=ifft2(T);
L2=real(L1);
figure, imshow(L2,[]);

end
```

The discrete Fourier transform of a function (image) f(x,y) of size MxN is given by the equation:

$$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}.$$

This expression is computed for values of u= 0 to M-1, and also for v=0 to N-1 .

The Fourier transform has the following shifting property:

$$f(x-x_0, y-y_0) \Leftrightarrow F(u,v)e^{-j2\pi(ux_0/M+vy_0/N)}.$$

where xo = counter1 and yo= counter2.

F(u,v)= k3(u,v) that is Fourier transform of image with the 4^{th} man only

M= 610 , N= 958.

Here two variables are taken (counter1, counter2) to control the movement of selected object i.e.4^{th} man. Initially their values are 1 which implies the initial position of the object which are incremented/decremented according to the desired direction of selected object. In this particular case counter 1 is decremented with a value -10 and counter2 is decremented with a value -20. This results in the movement in the diagonal direction.

Then another variable a is used to control the number of movements of the selected object. In the case of fourth man the number of movements are 9, so a is assigned the value 1 to 90 with step increment of 10 .

Within first for loop, we apply the shifting property of the Fourier transform using u and v for loops .The values of the u and v loop are assigned according to the size of the image taken which is 610 x 958.

```
for u = 1: 610
    for v = 1:958

  k4 (u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 +(v*counter2)/958)));

    end
    end
```

k4(u,v) gives the fourier transform of the image with the shifted object according to the value of counter1 and counter2.

We then calculate inverse Fourier transform of k4 which is stored in the matrix f1 and then the real part of f1 which is stored in f2.

Now the roifill(R2,c,r) command fills in the polygon specified by c and r, which are equal-length vectors containing the row-column coordinates of the pixels on vertices of the polygon. Here c and r, contain the row and column coordinates of the polygon containing 4^{th} man.

```
c = [682 754 950 935];
r = [227 569 586 210];
d = roifill(R2,c,r);
```

Now we calculate the Fourier transform k11 of the image d ,obtained after applying roifill command .

Then we add the k11 and k4 which is stored in the matrix T.

We then calculate inverse Fourier transform of T which is stored in the matrix L1 and then the real part of L1 is stored in matrix L2.

Using imshow(L2,[]) command we display the image with shifted object.
Now according to the value of a, the a loop executes 9 times which gives 7 movements of the selected object.

Figure 12. Image with 4th man shifted.

Step 14:

```
%For the ball
T2=imread('ballshift1.png');
T2=T2(34:643,92:1049);
figure,imshow(T2);

J = I;
% Turn the ball black
J(s(50).PixelIdxList) = 0;
figure,imshow(J);

% fourier transform of main image
k1=fft2(I);

% fourier transform the image with blackened ball
k2=fft2(J);

% fourier transform of image with the ball only
```

```
k3=k1-k2;
y3=ifft2(k3);
y4=real(y3);
```

The image obtained after shifting 4th man is saved as ballshift.png and thereafter image size is reduced to 610x958.
We assign the main image array I to J.

Labeled region 50 in the label matrix L corresponds to the ball.
So element 50 of the array s having field PixelIdxlist is assigned zero value.
p-element vector containing the linear indices of the pixels in the labeled region 50 in the label matrix L is assigned zero, which in turn turns the ball black.
K1 calculates the Fourier transform of main image I.

K2 calculates Fourier transform the image with 1st man blackened J.

K3 that is Fourier transform of image with the ball only is obtained by subtracting K2 from K1.
We then calculate inverse Fourier transform of k3 which is stored in the matrix y3 and then the real part of y3 stored in matrix y4.

Using imshow(y4,[]) command we display the image with ball only.

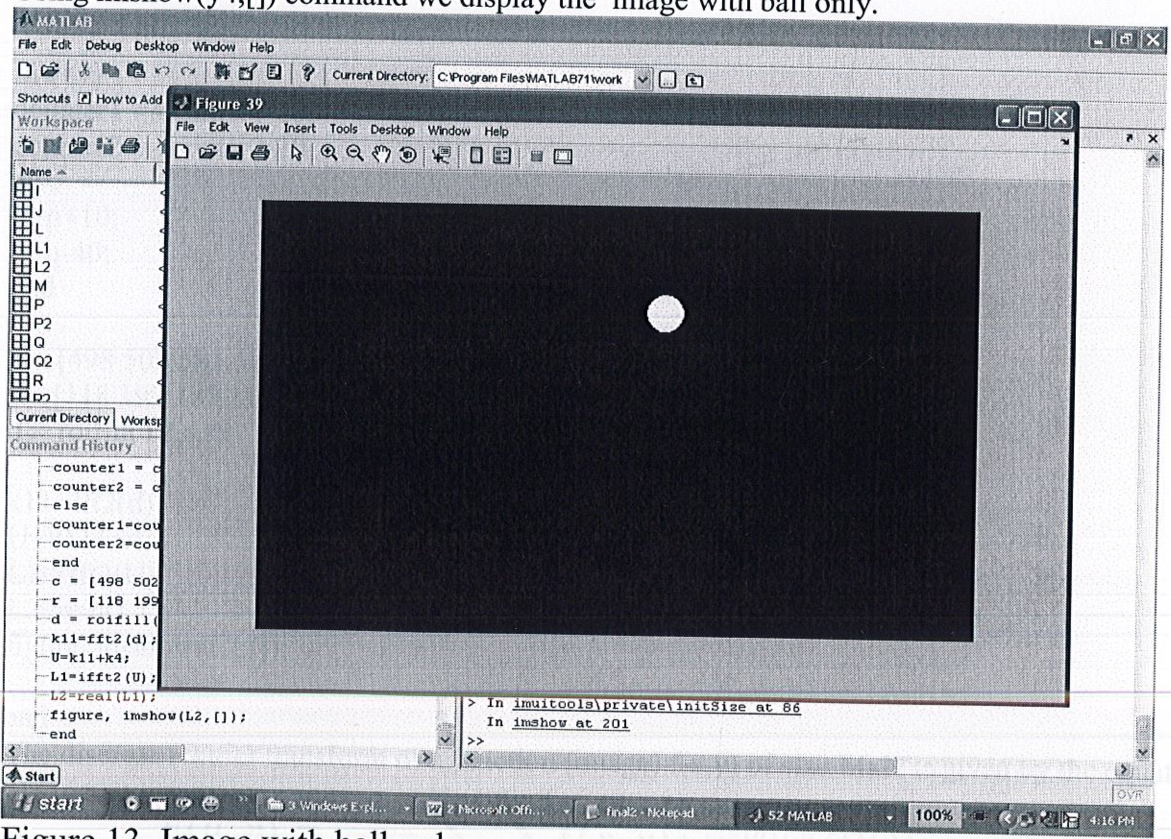


Figure 13. Image with ball only.

Step 15:

```
% Loop to display movements of ball
Counter1=1;
Counter2=1;
n=1;

for a = 1:10:120
  n=n+1;

    for u = 1: 610
    for v = 1:958

  k4(u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 + (v*counter2)/958 )));

    end
    end

f1=ifft2(k4);
f2=real(f1);

if(n<7)
counter1 = counter1-20;
counter2 = counter2-20;
else
p=p+10;
q=q-40;
end

c = [498 502 581 580];
r = [118 199 187 115];
d = roifill(T2,c,r);

k11=fft2(d);
U=k11+k4;
L1=ifft2(U);
L2=real(L1);
figure, imshow(L2,[]);

end
```

The discrete Fourier transform of a function (image) f(x,y) of size MxN is given by the equation:

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}.$$

This expression is computed for values of u= 0 to M-1, and also for v=0 to

N-1 .

The Fourier transform has the following shifting property:

$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M + vy_0/N)}$$

where xo = counter1 and yo= counter2.
F(u,v)= k3(u,v) that is Fourier transform of image with the 4th man only
M= 610 , N= 958.

Here two variables are taken (counter1, counter2) to control the movement of selected object i.e.4th man. Initially their values are 1 which implies the initial position of the object which are incremented/decremented according to the desired direction of selected object. In this particular case counter 1 is decremented with a value -10 and counter2 is decremented with a value -20. This results in the movement in the diagonal direction.

Then another variable a is used to control the number of movements of the selected object. In the case of fourth man the number of movements are 9, so a is assigned the value 1 to 90 with step increment of 10 .

Within first for loop, we apply the shifting property of the Fourier transform using u and v for loops .The values of the u and v loop are assigned according to the size of the image taken which is 610 x 958.

```
for u = 1: 610
    for v = 1:958

 k4 (u,v)=k3(u,v)*(exp(-j*2*3.14*( (u*(counter1))/610 +(v*counter2)/958)));

    end
    end
```

k4(u,v) gives the fourier transform of the image with the shifted object according to the value of counter1 and counter2.

We then calculate inverse Fourier transform of k4 which is stored in the matrix f1 and then the real part of f1 which is stored in f2.
Now the roifill(R2,c,r) command fills in the polygon specified by c and r, which are equal-length vectors containing the row-column coordinates of the pixels on vertices of the polygon.
Here c and r, contain the row and column coordinates of the polygon containing 4th man.

```
c = [498 502 581 580];
r = [118 199 187 115];
d = roifill(T2,c,r);
```

Now we calculate the Fourier transform k11 of the image d ,obtained after applying roifill command .

Then we add the k11 and k4 which is stored in the matrix U.
We then calculate inverse Fourier transform of U which is stored in the matrix L1 and then the real part of L1 is stored in matrix L2.

Using imshow(L2,[]) command we display the image with shifted object.
Now according to the value of a, the a loop executes 9 times which gives 9 movements of the selected object.

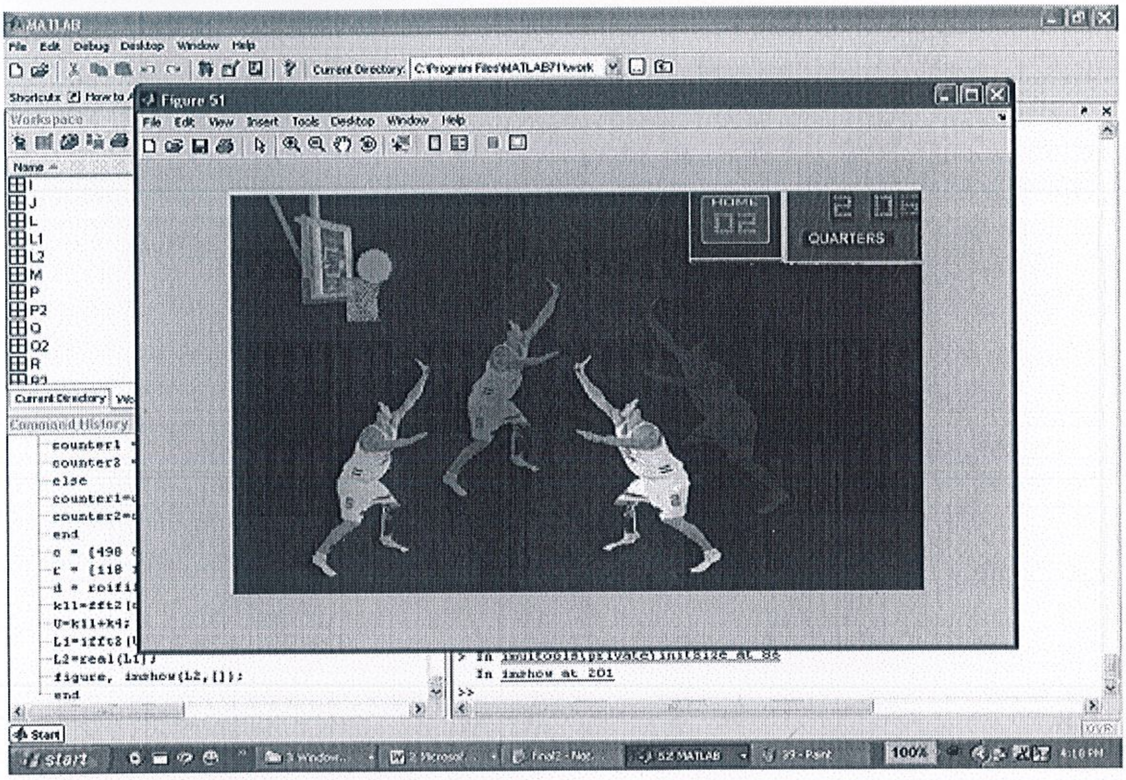


Figure 14. Image with ball shifted.

CONCLUSION

In this project shifting of an object in an image in frequency domain is done. Here we have labeled the required object and then extracted it from the main image. Thereafter shifting of the extracted object is done using shifting property of the Fourier Transform.

Future work done can be marking of all objects with the required boundary in the image , and thereby extracting them from the image. Animation can also be shown for the various steps of the shifting.

BIBLOGRAPHY

Websites – **www.google.co.in**
http://blogs.mathworks.com/steve/
http://www.dspguide.com/ch24.htm
http://www.qi.tnw.tudelft.nl/Courses/FIP/noframes/fip--2.html

Text Book – Digital Image Processing by Rafael C. Gonzalez and Richard E. Woods