NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Laboratory 1 Report

Contrast Stretching + Histogram Equalisation + Linear Spatial Filtering + Median Filtering + Suppressing Noise Interference Patterns + Undoing Perspective Distortion of Planar Surface

Nathan Soh

2.1 Contrast Stretching

a. Input the image into a MATLAB Matrix and convert it into a grayscale image:

Code:

```
Pc = imread('mrttrainbland.jpeg');
whos Pc
P = rgb2gray(Pc);
whos P
```

Result:

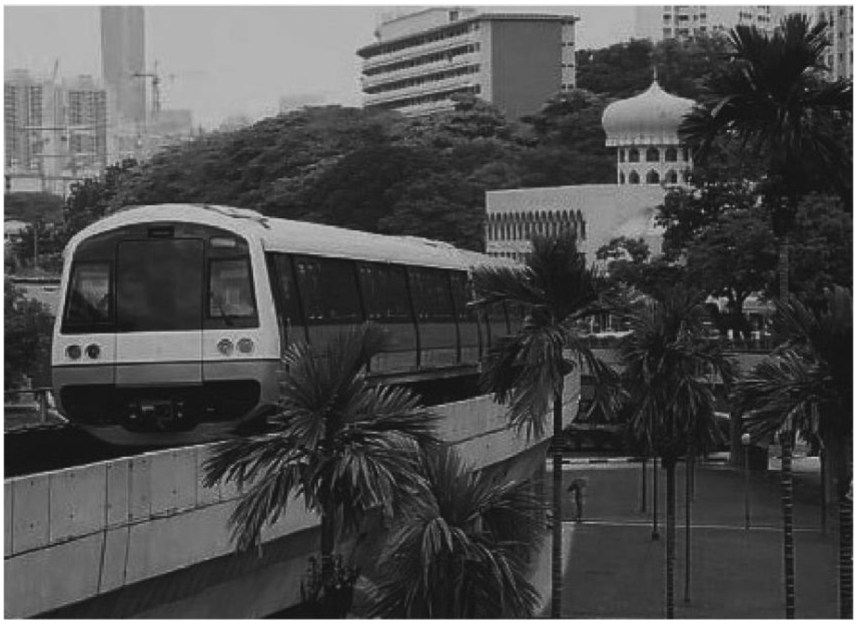
Pc:

```
 Name         Size                Bytes  Class     Attributes

 Pc          320x443x3           425280  uint8
```

P:

```
 Name         Size                Bytes  Class     Attributes

 P          320x443              141760  uint8
```

b. View the image using **imshow:**



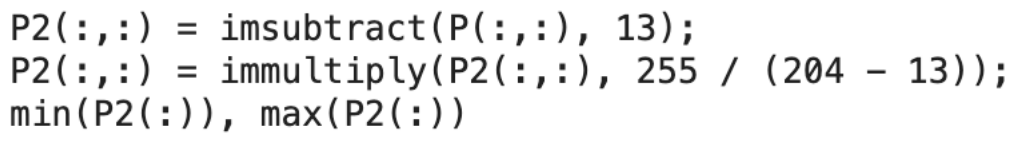
c. Check the minimum and maximum intensities present in the image:

Minimum Intensity: 13

Maximum Intensity: 204

d. Write two lines of MATLAB code to do contrast stretching:

Code:



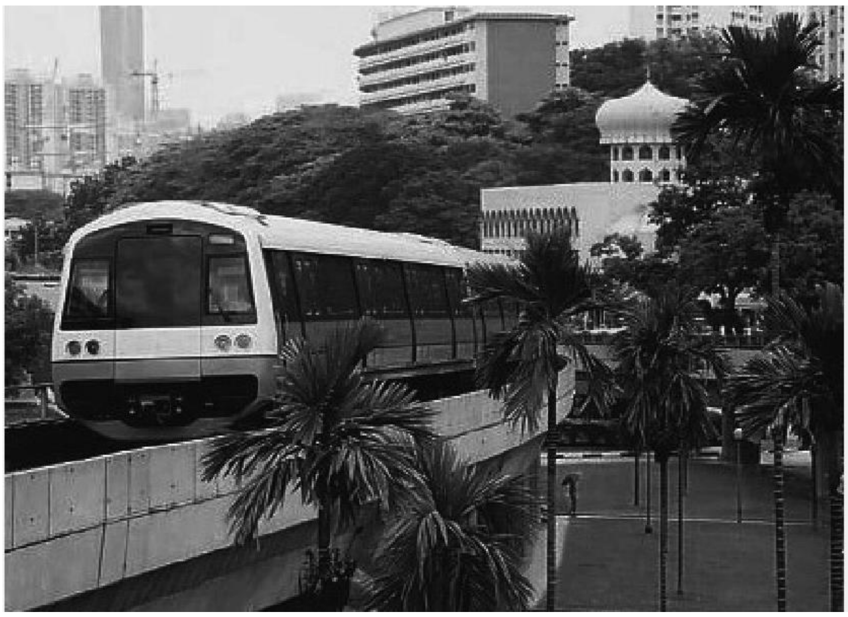
```
P2(:,:) = imsubtract(P(:,:), 13);
P2(:,:) = immultiply(P2(:,:), 255 / (204 - 13));
min(P2(:)), max(P2(:))
```

Result:

Minimum Intensity: 0

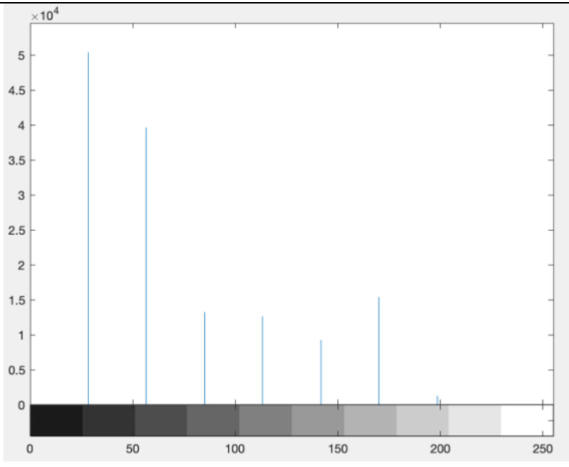
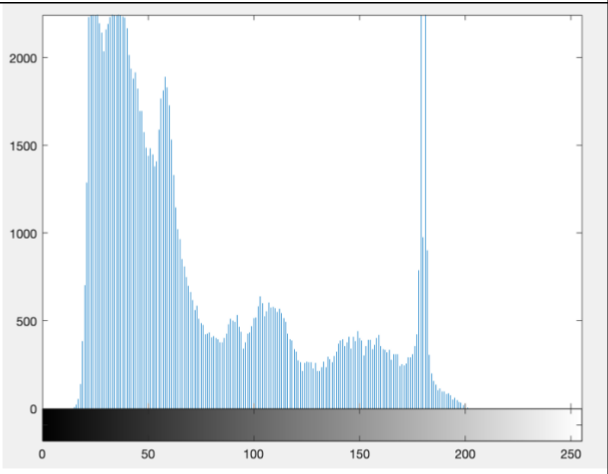
Maximum Intensity: 255

e. Redisplay the image after contrast stretching:



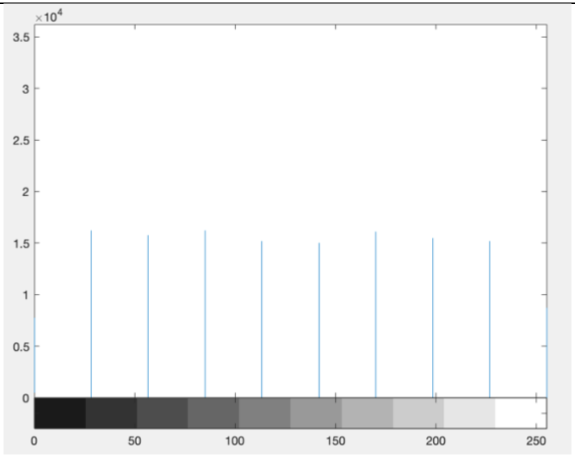
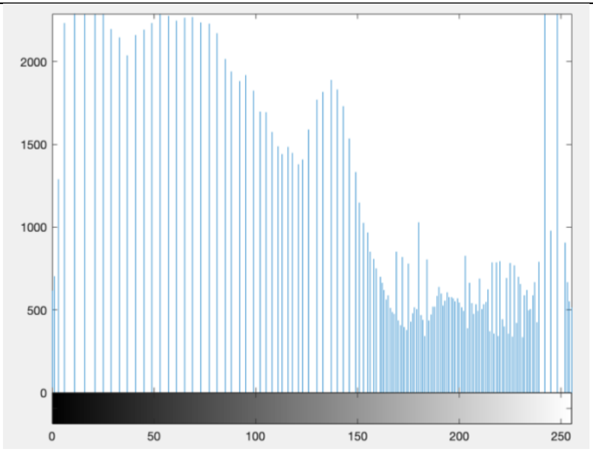
2.2 Histogram Equalization

a. Display the image intensity histogram of P

10 Bins	256 Bins
	

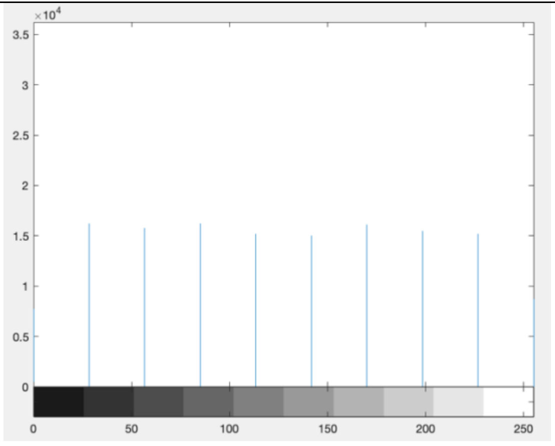
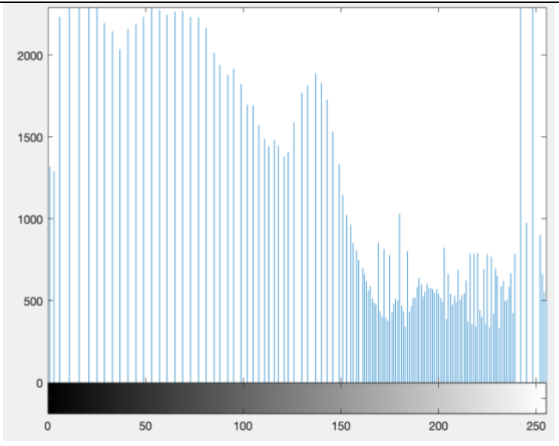
From the images, you can see the difference in number of bins that the pixels are grouped in, and hence the image with 10 bins shows a more generalised overview of how the pixels are grouped, while the 256 bins image gives a more comprehensive overview of how the pixels are grouped. However, the trend of the frequencies on both histograms are similar.

b. Carry out histogram equalisation and redisplay the histograms:

10 Bins	256 Bins
	

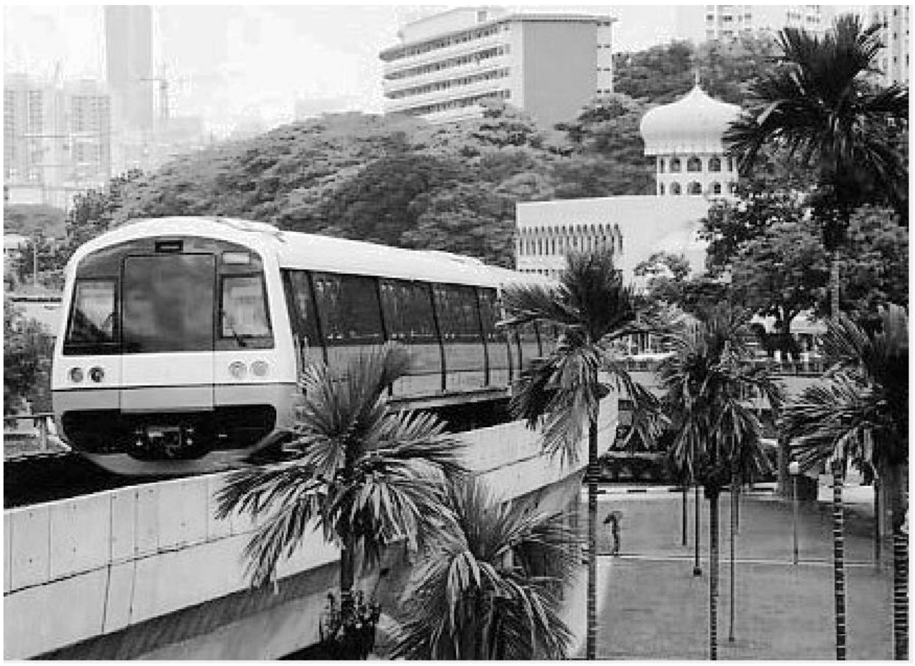
From the 2 bins, the 10 bins histogram seem to be uniform while the 256 bin histogram does not seem to be uniform. However, as compared to the latter histograms, the pixels in these histograms are more equally distributed between the bins.

c. Rerun the histogram equalisation:

10 Bins	256 Bins
	

The histograms did not become more uniform. In fact, the histograms are identical. This is because histogram equalisation is idempotent where the values in the histogram does not change when it is multiplied by itself.

Image after Histogram Equalisation:

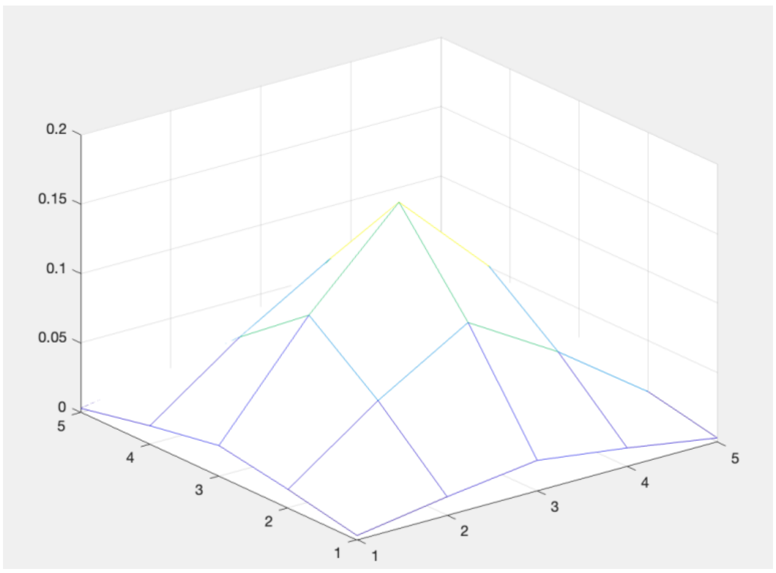


2.3 Linear Spatial Filtering

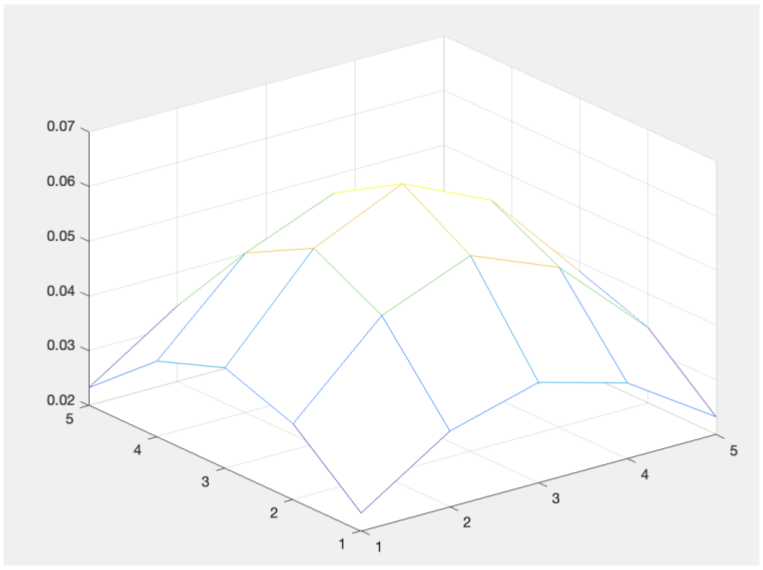
a. Generate the Gaussian Filters:

```
N = 5;
sigma1 = 1;
sigma2 = 2;
ind = -floor(N/2) : floor(N/2);
[X Y] = meshgrid(ind, ind);

%// Create Gaussian Mask
h1 = exp(-(X.^2 + Y.^2) / (2*sigma1*sigma1))/(2*pi*sigma1*sigma1);
h2 = exp(-(X.^2 + Y.^2) / (2*sigma2*sigma2))/(2*pi*sigma2*sigma2);
%// Normalize so that total area (sum of all weights) is 1
h1 = h1 / sum(h1(:));
h2 = h2 / sum(h2(:));
```

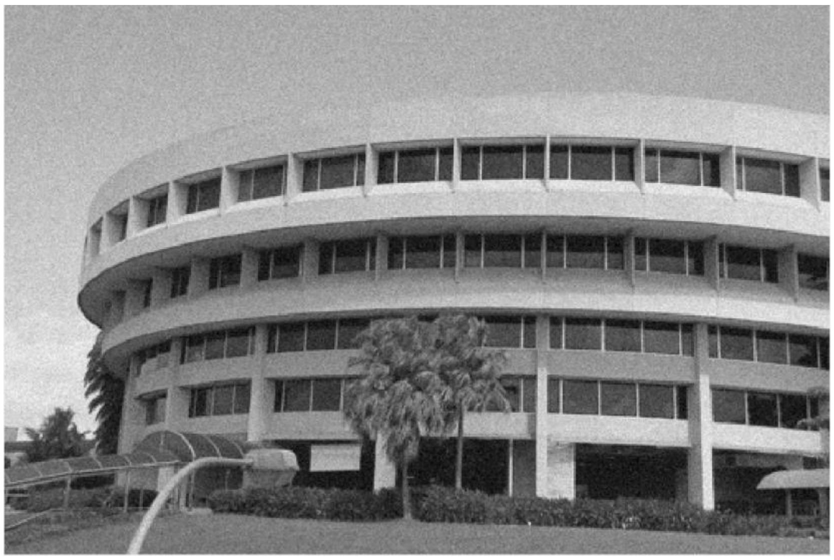
1. Y and X-dimensions are 5 and σ = 1.0



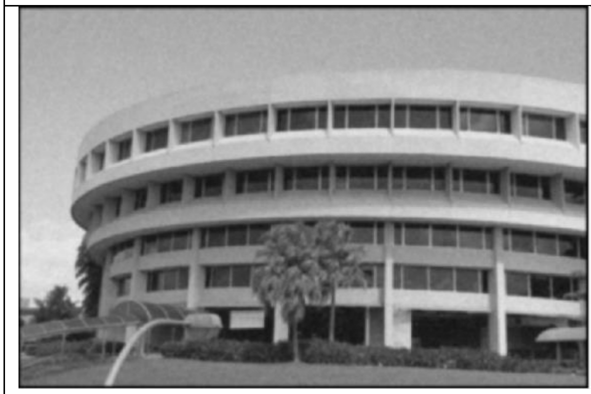
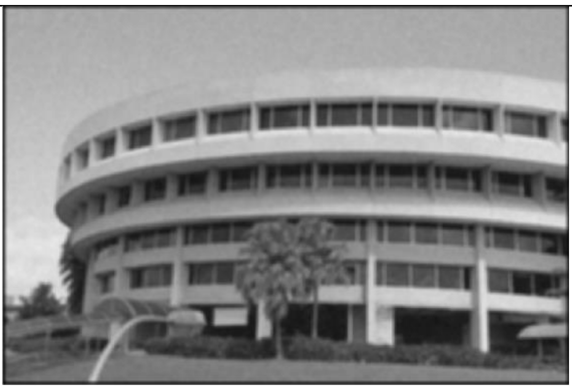
2. Y and X-dimensions are 5 and σ = 2.0



b. Display the ‘ntu-gn.jpg’ image:

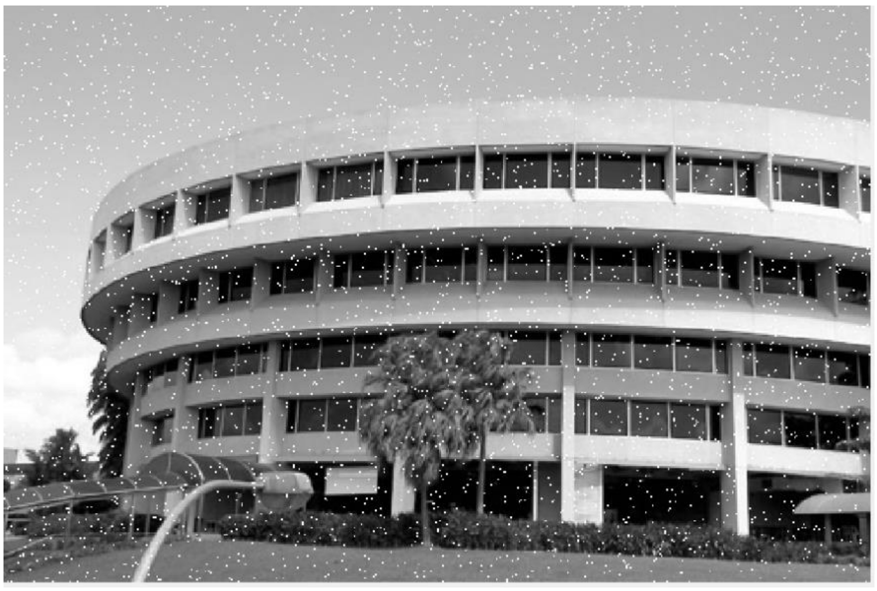


c. Filter the image using the created filters:

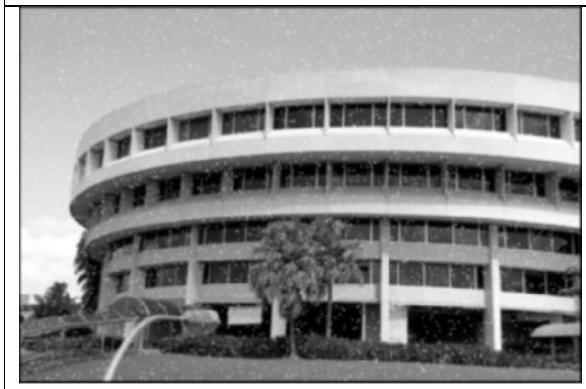
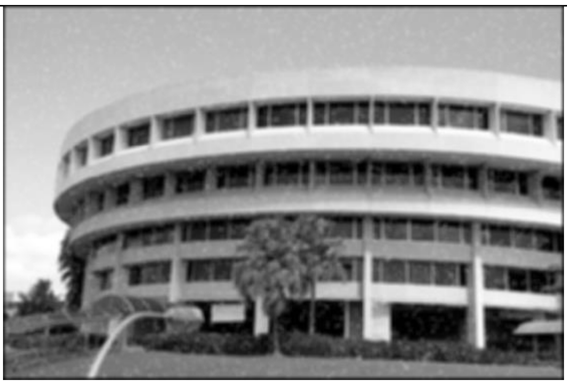
σ = 1.0	σ = 2.0
	

The second filter does reduce more noise as compared to the first filter. However, the first filter retains more detail than the second filter. Therefore, the higher the σ-value, the greater the noise removal, but less details will be retained. This shows that filtering results in information loss.

d. Display the ‘ntu-sp.jpg’ image:



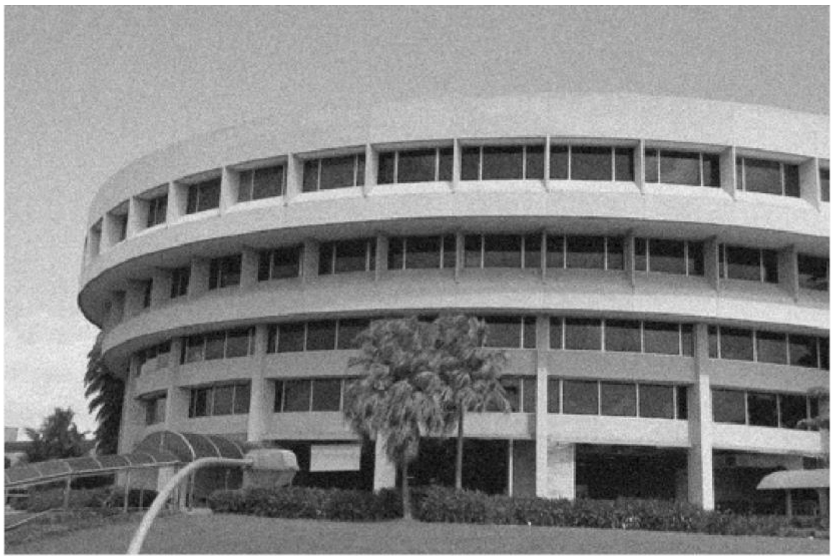
e. Filter this image using the created filters:

σ = 1.0	σ = 2.0
	

The filters are better at filtering gaussian noise rather than speckle noise. The speckle noise is still apparent in the images even after filtering. This is because the Gaussian Averaging Filter gives weight to all surrounding pixels but the speckle noise has pixels with intensities very different from the other pixels.

2.4 Median Filtering

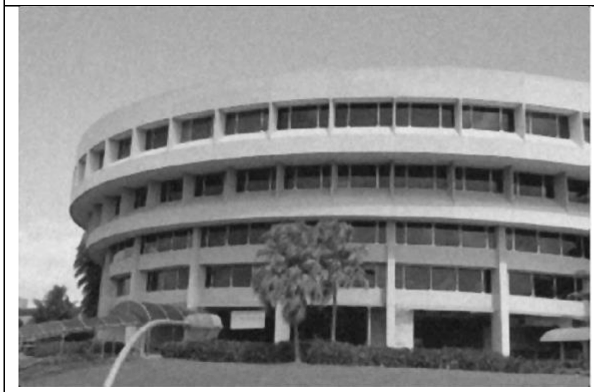
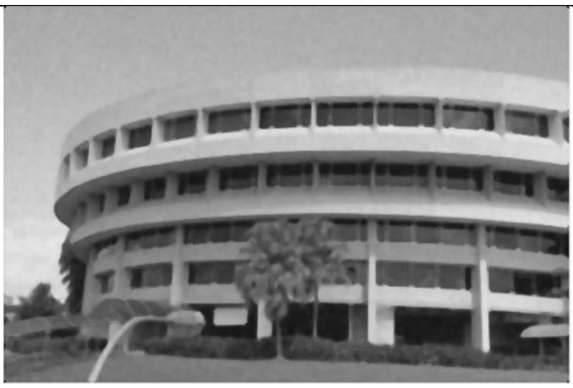
b. Display the ‘ntu-gn.jpg’ image:



c. Filter the image using median filtering:

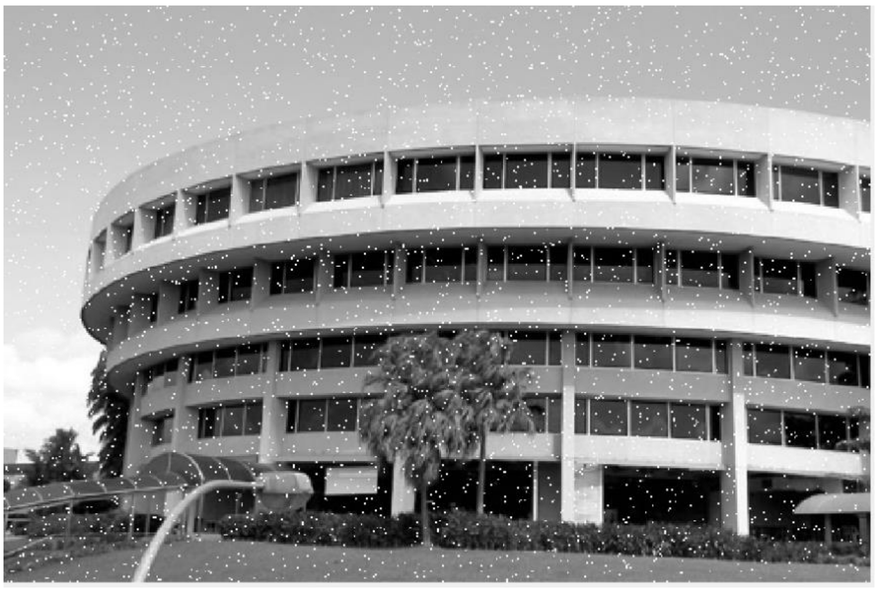
Code:

```
g1 = medfilt2(Gn, [3 3]);
g2 = medfilt2(Gn, [5 5]);
```

3 x 3	5 x 5
	

Median filtering does not perform very well in removing additive Gaussian noise as the noise is still apparent in both the filtered images. It is also observed that a greater neighbourhood size results in loss of information as the 5 x 5 image is less detailed than the 3 x 3 image, which is also less detailed than the original image.

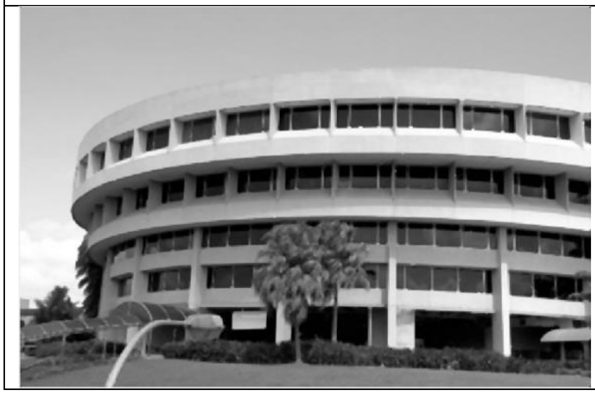
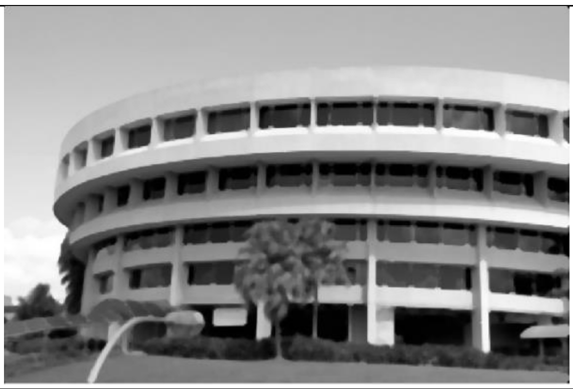
d. Display the ‘ntu-sn.jpg’ image:



e. Filter the image using Median Filtering:

Code:

```
s1 = medfilt2(Sp, [3 3]);
s2 = medfilt2(Sp, [5 5]);
```

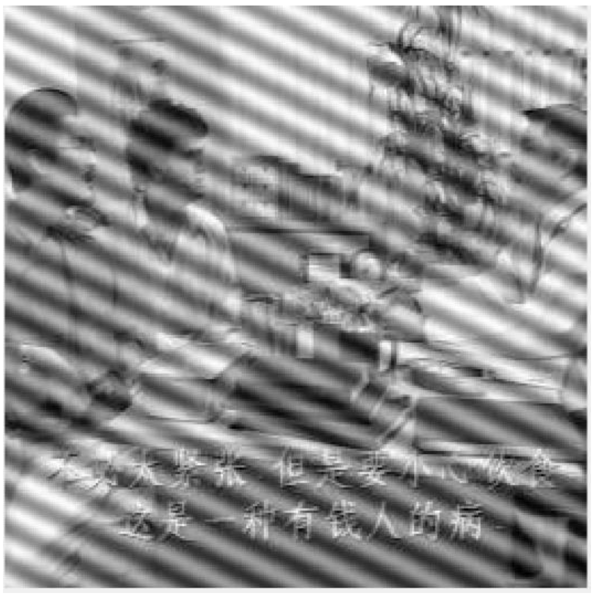
3 x 3	5 x 5
	

Median Filtering is more effective at removing speckle noise from an image, but loses detail in the image as a trade-off. Similarly to the filtering on the image with the additive gaussian noise, a higher neighbourhood size results in more details lost.

All in all, from this experiment, we can clearly see that gaussian filtering is more effective in removing additive gaussian noise from an image, while median filtering is more effective in removing speckle noise from an image.

2.5 Suppressing Noise Interference Patterns

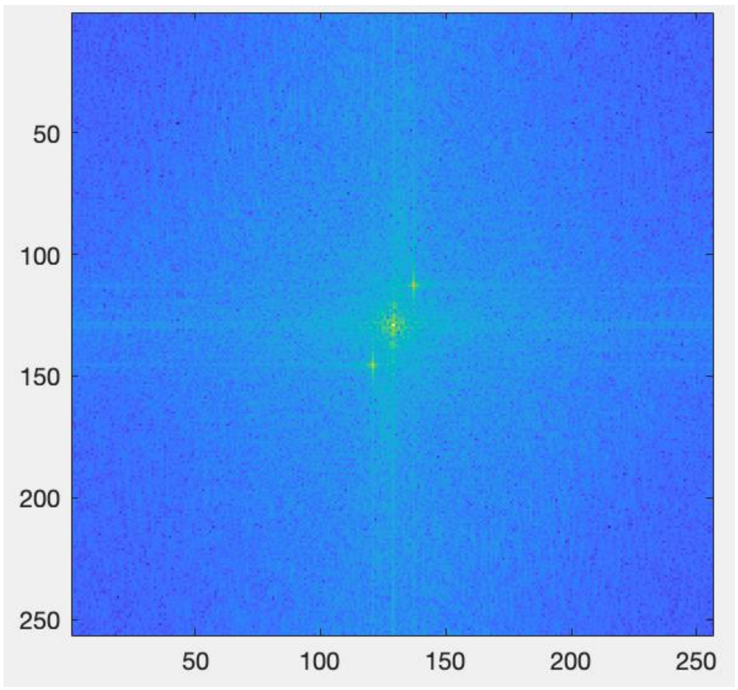
a. Display the ‘pck-int.jpg’ image:



b. Obtain the Fourier Transform F of the image:

Code:

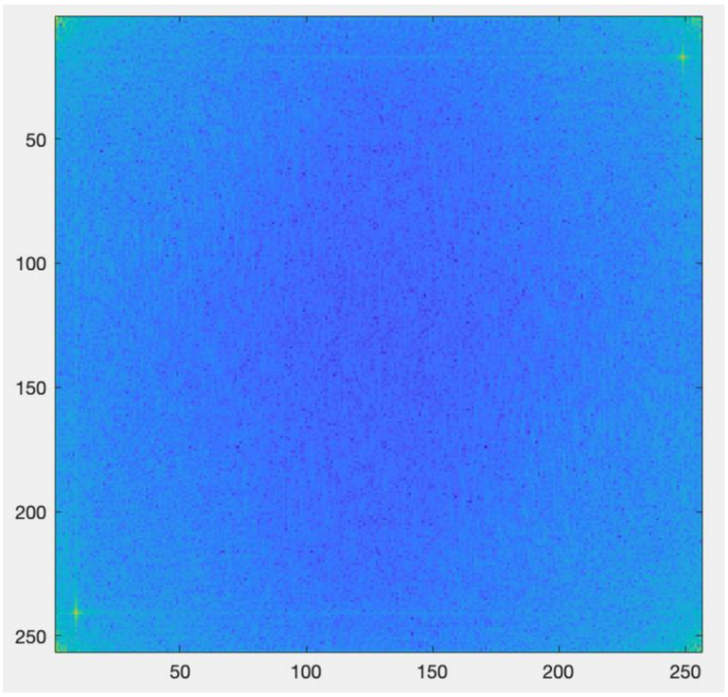
```
F = fft2(I);
S = abs(F);
imagesc(fftshift(S.^0.1));
colormap('default');
```



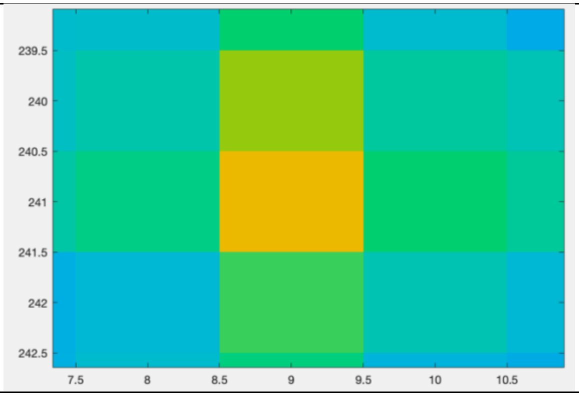
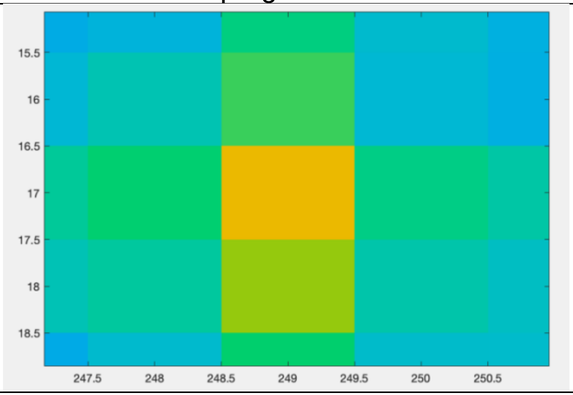
c. Redisplay the power spectrum without fftshift:

Code:

```
imagesc(S.^0.1)
colormap('default')
```



From the figure above, we can see that there are 2 peaks – one at the bottom left corner and one at the top right corner. To measure the coordinates of the peaks, we will zoom in to read the coordinates of the peaks

Bottom left corner	Top right corner
	

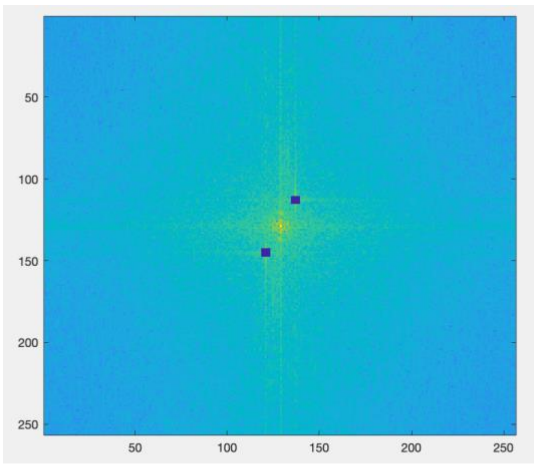
From the zoomed in image, we can see that the peaks are at:

1. x = [240.5, 241.5], y = [8.5, 9.5] => (241, 9)
2. x = [16.5, 17.5], y = [248.5, 249.5] => (17, 249)

d. Set to zero the 5x5 neighbourhood elements at locations corresponding to the peaks in the Fourier Transform F.

Code:
```
x1 = 241; y1 = 9;
x2 = 17; y2 = 249;
F(x1-2:x1+2, y1-2:y1+2) = 0;
F(x2-2:x2+2, y2-2:y2+2) = 0;
S = abs(F);
imagesc(fftshift(S.^0.1));
colormap('default');
```

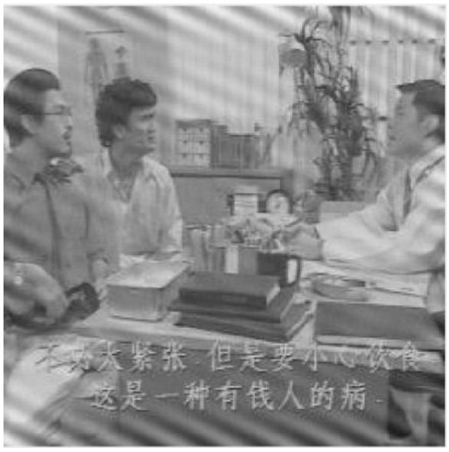
New Power Spectrum:



e. Compute the inverse Fourier Transform using ifft2:

Code:
```
result = uint8(ifft2(F));
imshow(result)
```

Resultant Image:



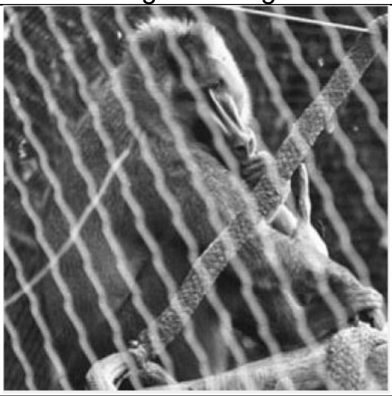
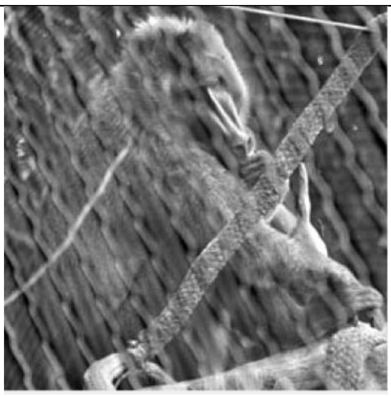
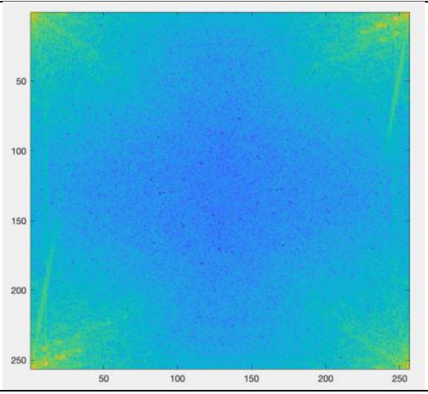
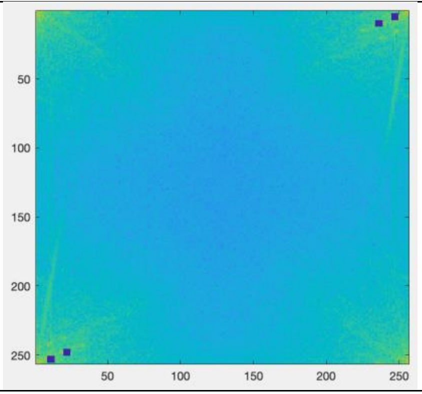
The resultant image has a lot of noise removed. In the spectral domain, the noise corresponds to the high frequency peaks and since these peaks are set to 0, the patterns associated with these frequencies are also removed from the spatial image.

However not all noise is removed from the image as not all the noise in the image corresponds to those peaks. To improve the result, we can try find other peaks within the spectral domain and set them to 0 as well.

e. Remove the fence from the ‘primate-caged.jpg’ image:

Code:
```
I2 = imread("primatecaged.jpeg");
I2 = rgb2gray(I2);
imshow(I2);

F = fft2(I2);
S = abs(F);
imagesc(S.^0.0001);
colormap('default')

x1 = 5; y1 = 247;
x2 = 253; y2 = 11;
x3 = 10; y3 = 236;
x4 = 248; y4 = 22;


F(x1-2:x1+2, y1-2:y1+2) = 0;
F(x2-2:x2+2, y2-2:y2+2) = 0;
F(x3-2:x3+2, y3-2:y3+2) = 0;
F(x4-2:x4+2, y4-2:y4+2) = 0;

S = abs(F);
imagesc(S.^0.1);
colormap('default')

result = uint8(ifft2(F));
imshow(result)
```

Result:

Original Image	After
	
	

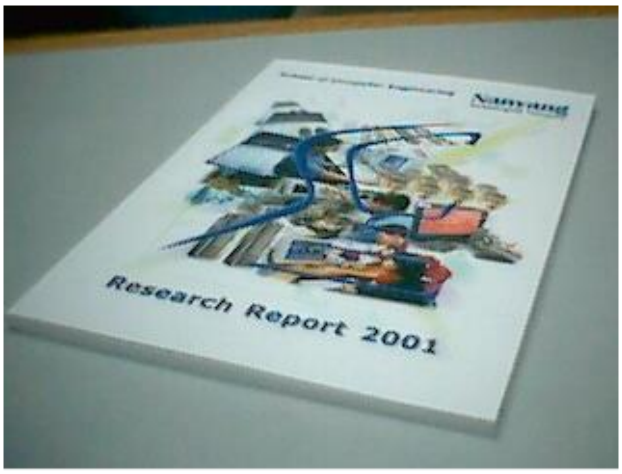
4 peaks were identified in the spectral domain and to free the primate, we would have to treat the fence as noise and remove it. We consider that the fence corresponds to the peaks in the spectral domain and the coordinates of the peaks are as follow:

1. x1, y1 = (5, 247)
2. x2, y2 = (253, 11)
3. x3, y3 = (10, 236)
4. x4, y4 = (248, 22)

It is observed that even after removing the peaks, the cage is not fully removed from the image. This could be due to the fact that the cage is not exactly a noise pattern and it does not fully correspond to the peaks in the spectral domain. To remove the fence, a more complex filter may be needed.

2.6 Undoing Perspective Distortion of Planar Surface

a. Download and Display the ‘book.jpg’ image



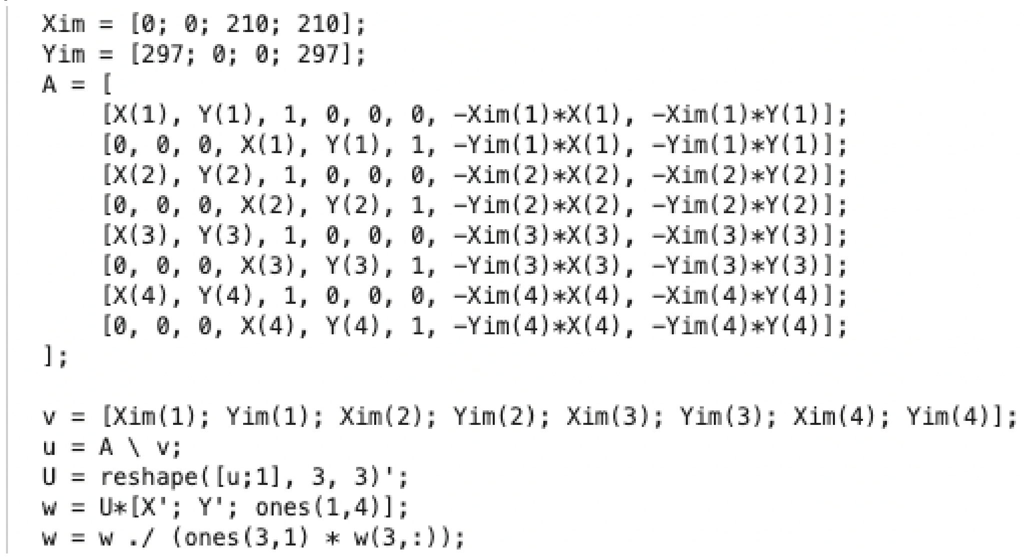
b. Obtain the 4 corner coordinates

Code:
```
[X Y] = ginput(4);
[X Y]
```

Result:
```
ans =

    5.0000  159.0000
  144.0000   28.0000
  308.0000   48.0000
  258.0000  216.0000
```

c. Set up the matrices required to estimate the projective transformation

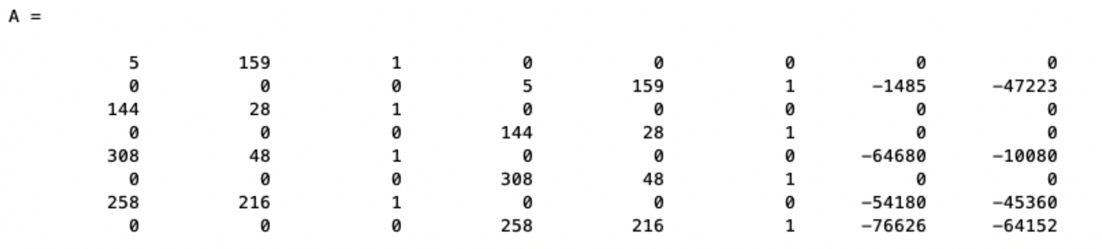
Code:



```
Xim = [0; 0; 210; 210];
Yim = [297; 0; 0; 297];
A = [
    [X(1), Y(1), 1, 0, 0, 0, -Xim(1)*X(1), -Xim(1)*Y(1)];
    [0, 0, 0, X(1), Y(1), 1, -Yim(1)*X(1), -Yim(1)*Y(1)];
    [X(2), Y(2), 1, 0, 0, 0, -Xim(2)*X(2), -Xim(2)*Y(2)];
    [0, 0, 0, X(2), Y(2), 1, -Yim(2)*X(2), -Yim(2)*Y(2)];
    [X(3), Y(3), 1, 0, 0, 0, -Xim(3)*X(3), -Xim(3)*Y(3)];
    [0, 0, 0, X(3), Y(3), 1, -Yim(3)*X(3), -Yim(3)*Y(3)];
    [X(4), Y(4), 1, 0, 0, 0, -Xim(4)*X(4), -Xim(4)*Y(4)];
    [0, 0, 0, X(4), Y(4), 1, -Yim(4)*X(4), -Yim(4)*Y(4)];
];

v = [Xim(1); Yim(1); Xim(2); Yim(2); Xim(3); Yim(3); Xim(4); Yim(4)];
u = A \ v;
U = reshape([u;1], 3, 3)';
w = U*[X'; Y'; ones(1,4)];
w = w ./ (ones(3,1) * w(3,:));
```

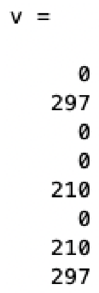
Results:

A:



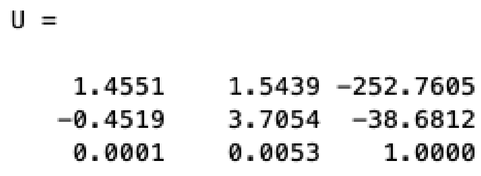
```
A =

       5       159         1         0         0         0         0         0
       0         0         0         5       159         1     -1485    -47223
     144        28         1         0         0         0         0         0
       0         0         0       144        28         1         0         0
     308        48         1         0         0         0    -64680    -10080
       0         0         0       308        48         1         0         0
     258       216         1         0         0         0    -54180    -45360
       0         0         0       258       216         1    -76626    -64152
```

v:



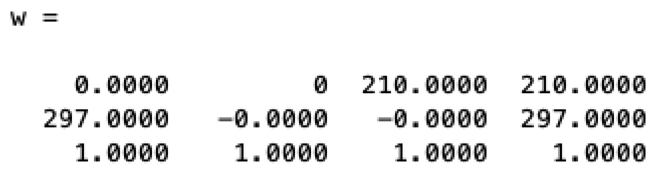
```
v =

     0
   297
     0
     0
   210
     0
   210
   297
```

U:



```
U =

    1.4551    1.5439 -252.7605
   -0.4519    3.7054  -38.6812
    0.0001    0.0053    1.0000
```

w:



```
w =

    0.0000         0  210.0000  210.0000
  297.0000   -0.0000   -0.0000  297.0000
    1.0000    1.0000    1.0000    1.0000
```

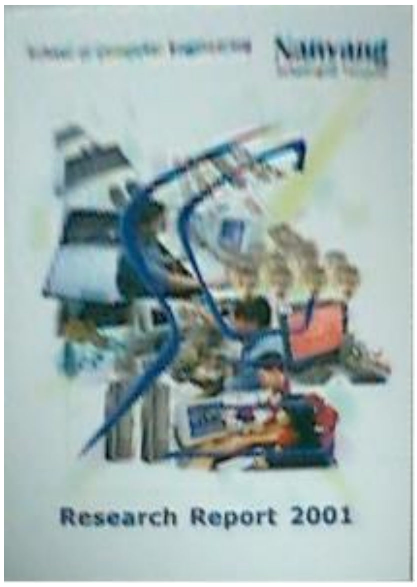
From the w matrix, we can see that the corners of the desired image is transformed correctly.

d. Warp the image

Code:

```
  T = maketform('projective', U');
  P2 = imtransform(P, T, 'XData', [0 210], 'YData', [0 297]);
```

e. Display the image



This is the output after transforming the image. The image proportions are exceeding expectations as the this book's proportions in the original image is quite distorted. It is also observed that the bottom half of the picture appears much clearer than the top half of the picture and this could be explained by the original picture where the bottom of the book in the original image is much clearer than the top half of the book so it is no surprise that the book still looks like that after the transformation.

All in all, the quality of the transformed image is very good with the accurate proportions and the relatively clear images and words.

f. Identify the ‘pink area’ between the ‘Nanyang’ and the ‘2001’ in the transformed image

Code:
```
[X Y] = ginput(4);

Xim = [0; 0; 297; 297];
Yim = [0; 210; 210; 0];
A = [
    [X(1), Y(1), 1, 0, 0, 0, -Xim(1)*X(1), -Xim(1)*Y(1)];
    [0, 0, 0, X(1), Y(1), 1, -Yim(1)*X(1), -Yim(1)*Y(1)];
    [X(2), Y(2), 1, 0, 0, 0, -Xim(2)*X(2), -Xim(2)*Y(2)];
    [0, 0, 0, X(2), Y(2), 1, -Yim(2)*X(2), -Yim(2)*Y(2)];
    [X(3), Y(3), 1, 0, 0, 0, -Xim(3)*X(3), -Xim(3)*Y(3)];
    [0, 0, 0, X(3), Y(3), 1, -Yim(3)*X(3), -Yim(3)*Y(3)];
    [X(4), Y(4), 1, 0, 0, 0, -Xim(4)*X(4), -Xim(4)*Y(4)];
    [0, 0, 0, X(4), Y(4), 1, -Yim(4)*X(4), -Yim(4)*Y(4)];
];

v = [Xim(1); Yim(1); Xim(2); Yim(2); Xim(3); Yim(3); Xim(4); Yim(4)];
u = A \ v;
U = reshape([u;1], 3, 3)';
w = U*[X'; Y'; ones(1,4)];
w = w ./ (ones(3,1) * w(3,:));

T = maketform('projective', U');
P3 = imtransform(P2, T, 'XData', [0 297], 'YData', [0 210]);
imshow(P3);
```

Result:

To identify the pink area, I performed a similar transformation to get the book, just that the orientation and dimension of the image is A4 (297 x 210) in landscape orientation instead of the portrait orientation. This is what is shown on the computer screen on the cover of the book.