Table of Contents


```
clc;
clear;
close all;
```

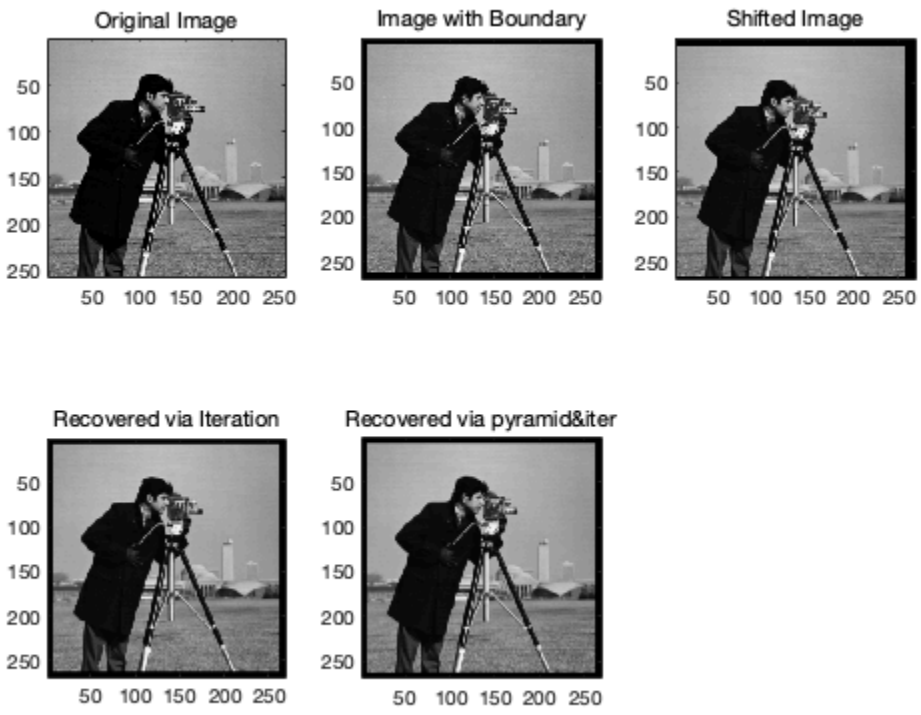
Part I

import the image

```
g0 = imread('cameraman_Original.tif');

% add the blank boundary
g_expanded = padarray(g0,[6 6],0,'both');

% translate the image
v_real = [-5.1,3.7];
g_trans = imtranslate(g_expanded,v_real);

% Optical Flow Recovering
[img_recov1, V1] = Optical_flow(g_expanded, g_trans, "T", 0, 100);

% Pyramid & iter
[img_recov2, V2] = pyramid_iter3(g_expanded, g_trans, 4, 100);
%
figure();
subplot(2,3,1);imagesc(g0);title('Original Image');
axis image;colormap gray;set(gca,'Visible','on');
subplot(2,3,2);imagesc(g_expanded);title('Image with Boundary');
axis image;colormap gray;set(gca,'Visible','on');
subplot(2,3,3);imagesc(g_trans);title('Shifted Image');
axis image;colormap gray;set(gca,'Visible','on');
subplot(2,3,4);imagesc(img_recov1);title('Recovered via Iteration');
axis image;colormap gray;set(gca,'Visible','on');
subplot(2,3,5);imagesc(img_recov2);title('Recovered via
 pyramid&iter');
axis image;colormap gray;set(gca,'Visible','on');

fprintf('The real motion is x:%8.4f; y: %8.4f\n',v_real(1),v_real(2));
fprintf('The Iter motion is x:%8.4f; y: %8.4f\n',V1(1),V1(2));
fprintf('The Pyra&iter motion is x:%8.4f; y: %8.4f\n',V2(1),V2(2));
```

```
The real motion is x: -5.1000; y:   3.7000
The Iter motion is x: -5.0992; y:   3.6998
```

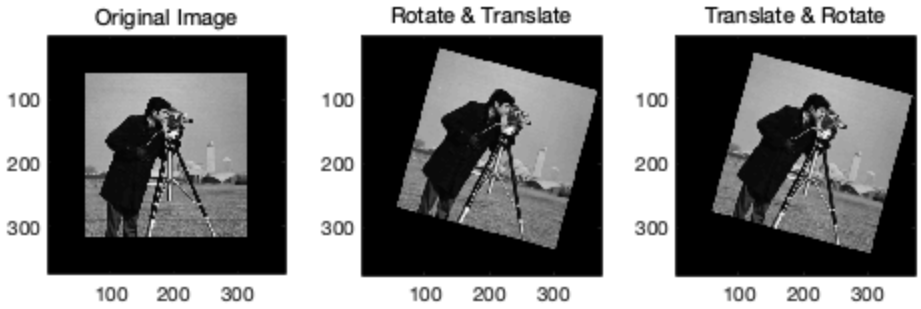
The Pyra&iter motion is x: -5.0999; y: 3.6999



Part II A&B

```
clc;
clear;
close all;

% wrap the image
g0 = imread('cameraman_Original.tif');
g0 = padarray(g0,[59 59],0,'both');
% translation and rotation
transx = 25;
transy = -10;
rot = 15;
% t & r in matlab format
M_T = [1,0,0; ...
       0,1,0; ...
       transx,transy,1];
M_R = [cosd(rot),sind(rot),0; ...
      -sind(rot),cosd(rot),0; ...
       0,0,1];
% using imref2d, move the original to the center
xWorldLimits = [1, size(g0,1)] - mean((1:size(g0,1)));
yWorldLimits = [1, size(g0,2)] - mean((1:size(g0,2)));
Imrf = imref2d(size(g0),xWorldLimits,yWorldLimits);
```

```
% rotate and then translate
Trans_rt = imwarp(g0,Imrf,affine2d(M_R*M_T),'OutputView',Imrf);
% translate and then rotate
Trans_tr = imwarp(g0,Imrf,affine2d(M_T*M_R),'OutputView',Imrf);

figure();
subplot(1,3,1);imagesc(g0);title('Original Image');
axis image;colormap gray;set(gca,'Visible','on');
subplot(1,3,2);imagesc(Trans_rt);title('Rotate & Translate');
axis image;colormap gray;set(gca,'Visible','on');
subplot(1,3,3);imagesc(Trans_tr);title('Translate & Rotate');
axis image;colormap gray;set(gca,'Visible','on');
```

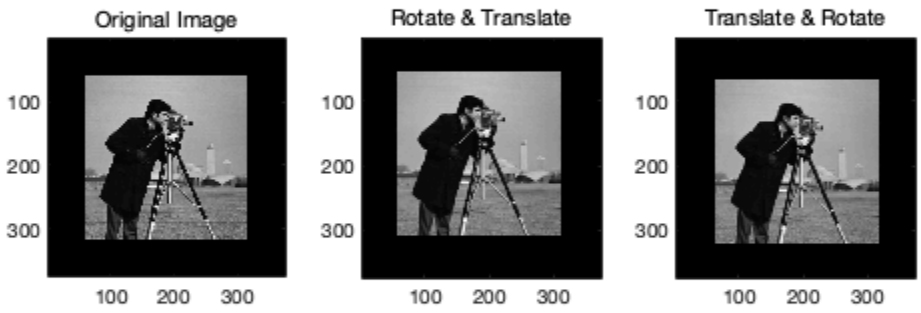


Part II C

since we know the motion, the inverse matrix should be

```
M_T_inv = [1,0,0; ...
       0,1,0; ...
       -transx,-transy,1];
M_R_inv = [cosd(-rot),sind(-rot),0; ...
      -sind(-rot),cosd(-rot),0; ...
       0,0,1];

% rotate and then translate
```

```
Trans_rt_inv =
 imwarp(Trans_rt,Imrf,affine2d(M_R_inv*M_T_inv),'OutputView',Imrf);
% translate and then rotate
Trans_tr_inv =
 imwarp(Trans_tr,Imrf,affine2d(M_T_inv*M_R_inv),'OutputView',Imrf);

figure();
subplot(1,3,1);imagesc(g0);title('Original Image');
axis image;colormap gray;set(gca,'Visible','on');
subplot(1,3,2);imagesc(Trans_rt_inv);title('Rotate & Translate');
axis image;colormap gray;set(gca,'Visible','on');
subplot(1,3,3);imagesc(Trans_tr_inv);title('Translate & Rotate');
axis image;colormap gray;set(gca,'Visible','on');
```



Published with MATLAB® R2020a