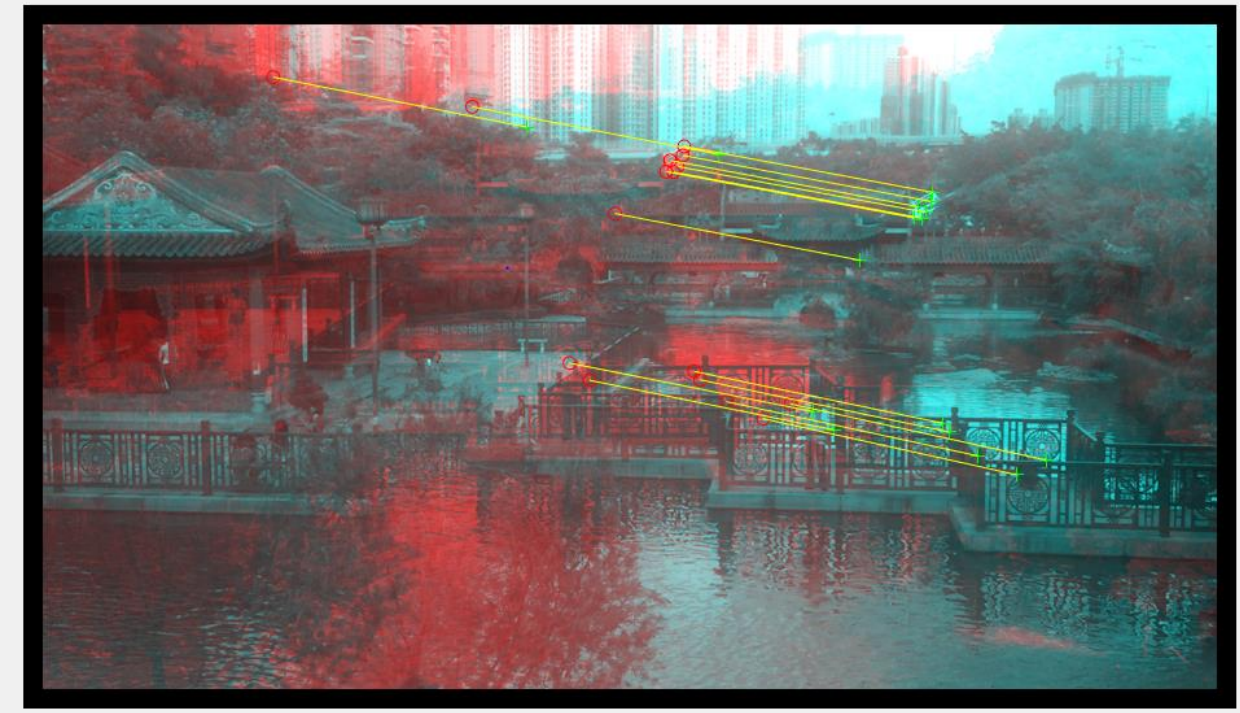
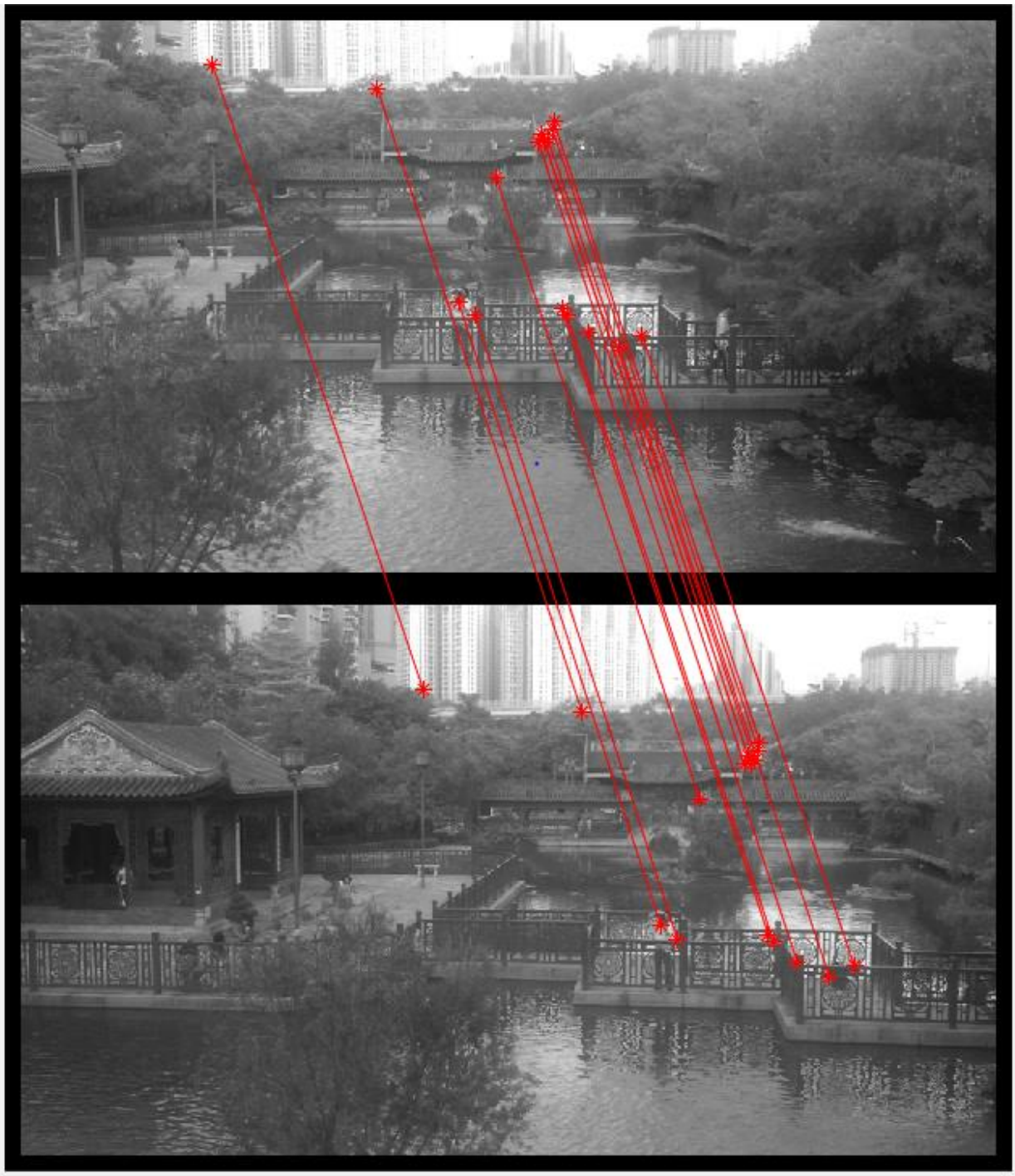
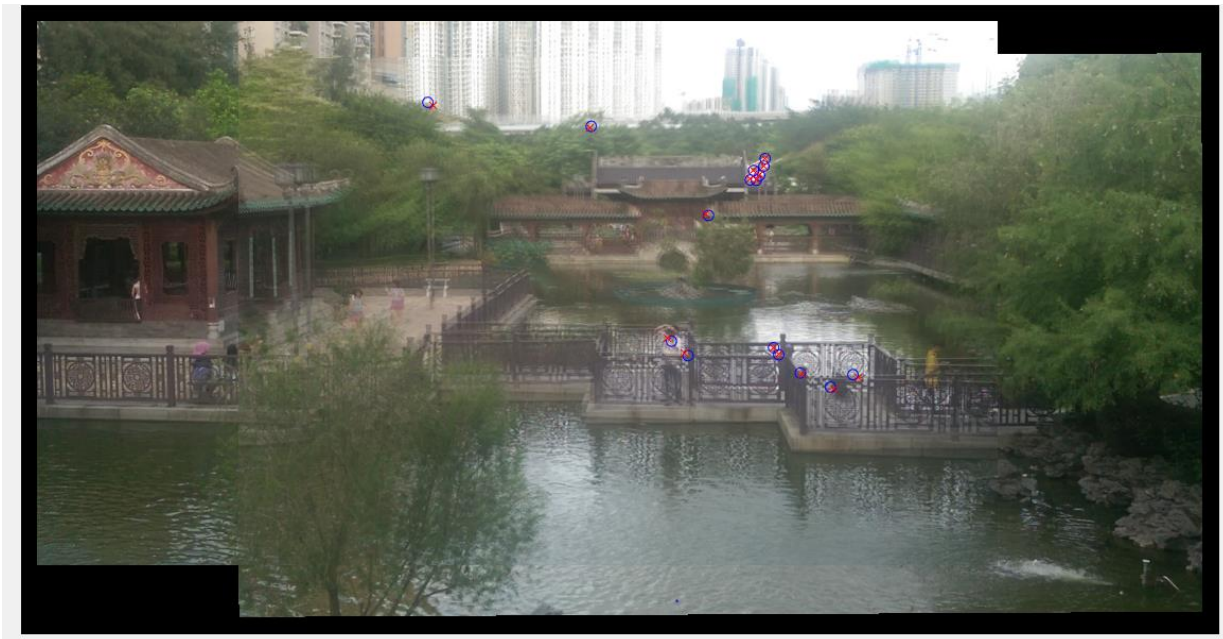
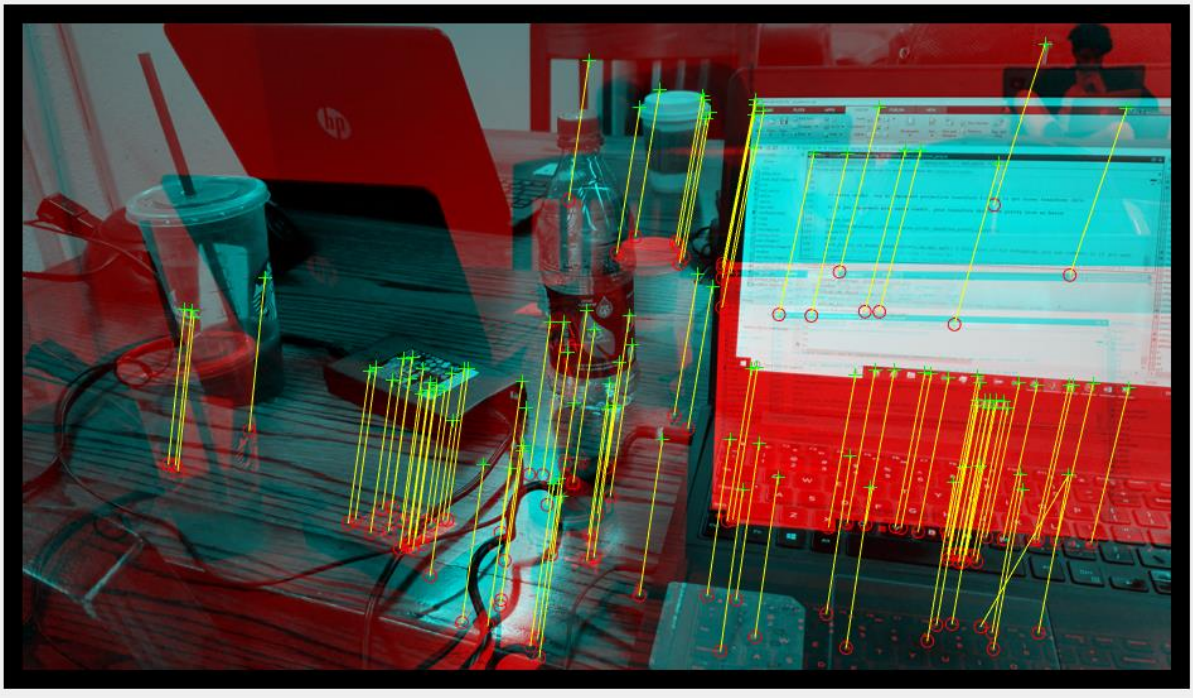
The given images:

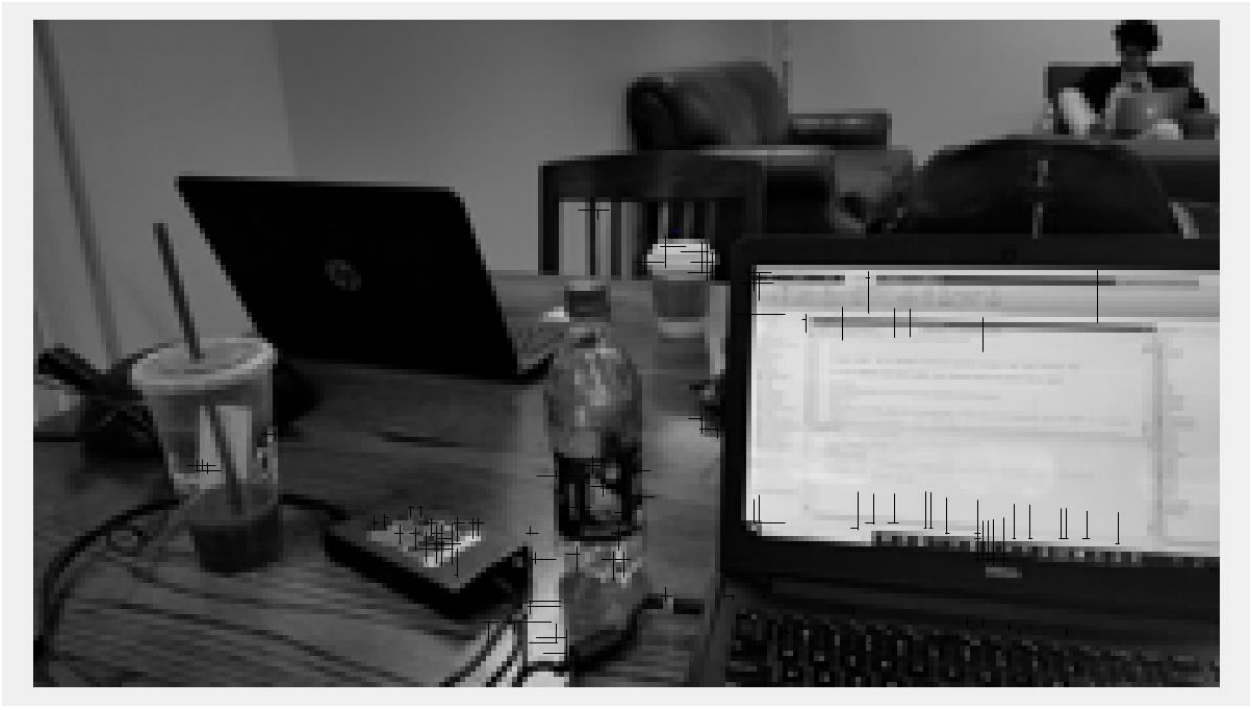


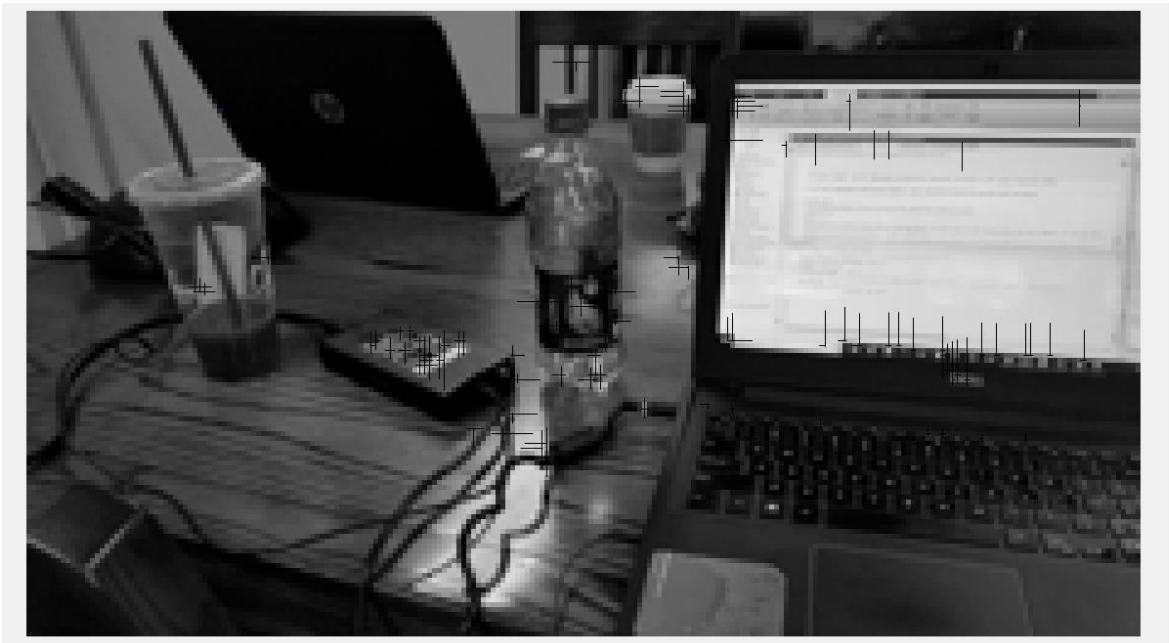


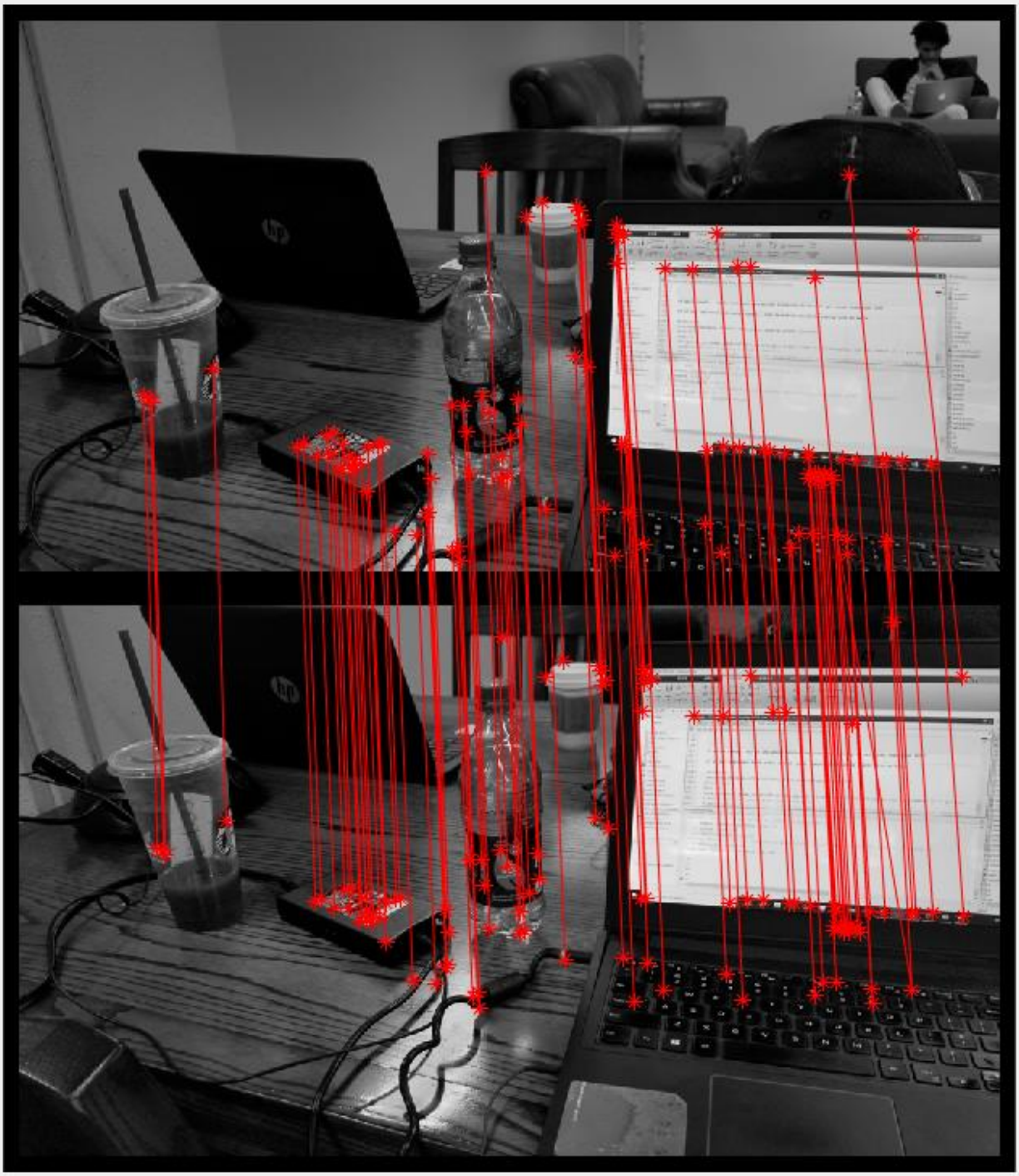


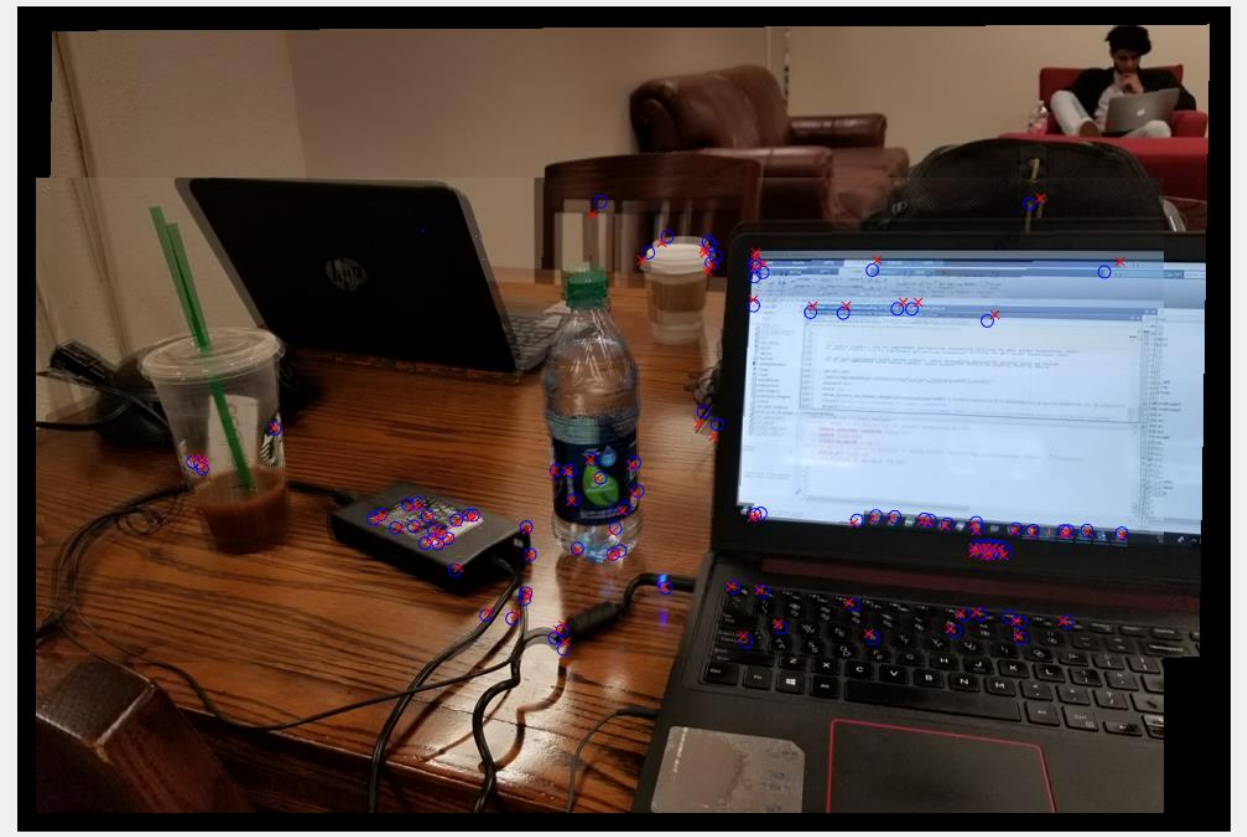
My own images for 20% bonus:











Source Code:

```
% run this script
%imnames={'i1.jpg','i2.jpg'};
imnames={'myPic1.jpg','myPic2.jpg'};

% resize image to reduce complexity
% pad image to avoid boundary problem
prevI1=preprocess_image(imread(imnames{1}));
prevI2=preprocess_image(imread(imnames{2}));
I1 = rgb2gray(prevI1);
I2 = rgb2gray(prevI2);

%I1 = rgb2gray(prevI1);
%I2 = rgb2gray(prevI2);

% Find the corners.
points1 = detectHarrisFeatures(I1);
points2 = detectHarrisFeatures(I2);

% Extract the neighborhood features.
```

```
[features1,valid_points1] = extractFeatures(I1,points1);
[features2,valid_points2] = extractFeatures(I2,points2);

% Match the features.
indexPairs = matchFeatures(features1,features2);

% Retrieve the locations of the corresponding points for
each image.
matchedPoints1 = valid_points1(indexPairs(:,1),:);
matchedPoints2 = valid_points2(indexPairs(:,2),:);

figure;
showMatchedFeatures(I1,I2,matchedPoints1,matchedPoints2);

%% Todo
% find match coordinates from I1 to I2
% You should comment the following line and create xy1 and
xy2 leveraging
% the HOG features created in HW4
% xy1 contains list of positions in I1 that match with
positions in I2
% (stored in xy2)
% xy1 and xy2 are homogenous coordinate, each column is one
point
% first coordinate of a point is the horizontal coordinate
of the image
% second coordinate of the point is the vertical coordinate
oimg1=imread('myPic1.jpg');
oimg1=rgb2gray(oimg1);
imgheight=128;
img1=imresize(oimg1,[imgheight,imgheight/size(oimg1,1)*size
(oimg1,2)]);

% list should hold matched points?
x = matchedPoints1.Location(1:end,2);
y = matchedPoints1.Location(1:end,1);

new_x = round(x.*128/542);
new_y = round(y.*227/936);
list=[new_x(:) new_y(:)];
tic; h1=myhog_list(img1,list,4,4); toc; % compute hog at
the locations in the list
visualize_hog_list(h1,list,img1);
xy1(1,:) = y;
```

```
xy1(2,:) = x;
xy1(3,:) = ones(1,size(x,1));

oimg2=imread('myPic2.jpg');
oimg2=rgb2gray(oimg2);
imgheight=128;
img2=imresize(oimg2,[imgheight,imgheight/size(oimg2,1)*size
(oimg2,2)]);
%[x,y]=ndgrid(10:10:120,10:10:220); % locations to compute
HOGs
x2 = matchedPoints2.Location(1:end,2);
y2 = matchedPoints2.Location(1:end,1);

new_x2 = round(x2.*128/542);
new_y2 = round(y2.*227/936);
list=[new_x2(:) new_y2(:)];

tic; h2=myhog_list(img2,list,4,4); toc; % compute hog at
the locations in the list
visualize_hog_list(h2,list,img2);

xy2(1,:) = y2;
xy2(2,:) = x2;
xy2(3,:) = ones(1,size(x2,1));
%load hw5.mat % you should comment this line, this load my
pre-matched pair list and transform
% check out xy1,xy2 and xx_sam
%% Don't need to change here
% visualize your points
% visualize_match(xy1,xy2,I1,I2);
visualize_match(xy1,xy2,I1,I2)

xx=affine_fit(xy1,xy2); % note that xy1 and xy2 are in
homogenous coordinates, each column is one point
% check out the predefined example in hw5.mat

[wholeImg,NewImage,offsets]=draw_align_image(xx,prevI1,prev
I2);
pause(0.5); % workaround for matlab bug with imshow and
hold
hold on;
show_points_on_drawn_image(offsets,xx,xy1,xy2); % this line
is for debugging, you can comment it if you want
title('Affine transform without ransac');
```

```
xlabel('Red cross are points in the second image; blue
circle are transformed points on the first image');

%% extra credit, try to implement ransac to get a better
fitting (50%)



%% extra credit, try to implement projective transform
fitting to get nicer transform (50%)

%% If you implement both extra credit, your transform
should be pretty nice as below

xx=xx_sam;
[wholeImg,NewImage,offsets]=draw_align_image(xx,prevI1,prev
I2);
pause(0.5);
hold on;
show_points_on_drawn_image(offsets,xx,xy1,xy2); % this line
is for debugging, you can comment it if you want
title('Projective transform fitting + ransac');
xlabel('Red cross are points in the second image; blue
circle are transformed points on the first image');
```