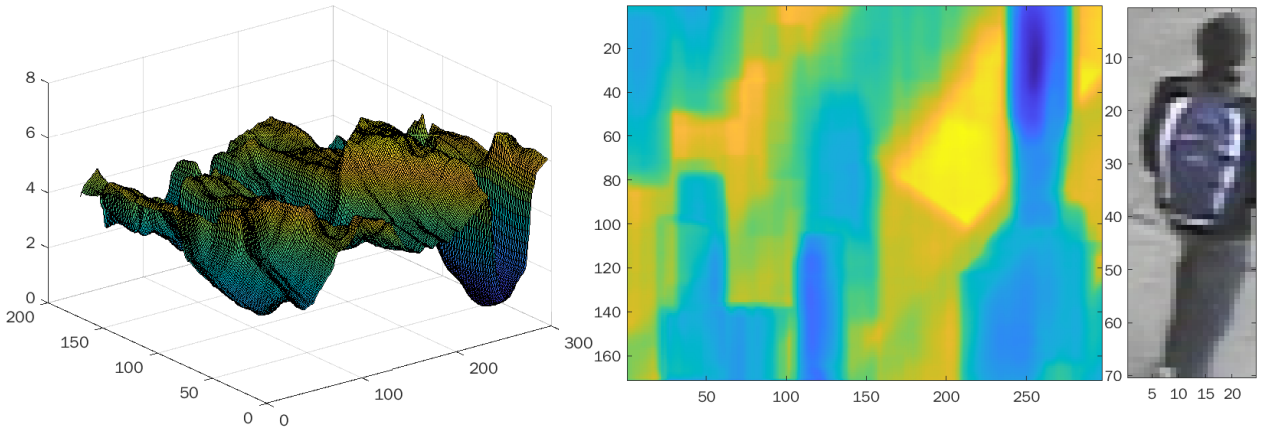
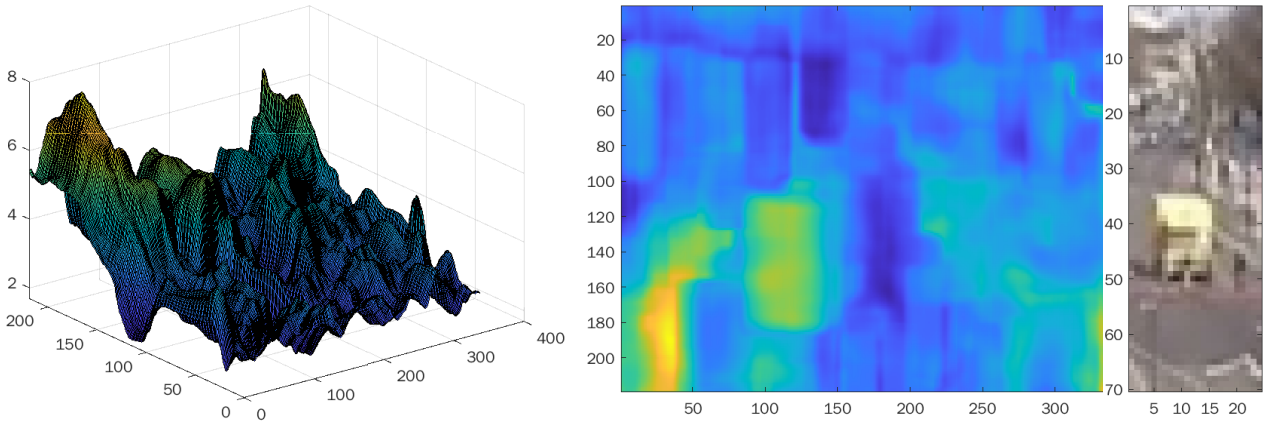
1) Use the covariance matching technique to find the correct match in the color image given on the WWW site (target.jpg). The model covariance matrix (of <x,y,R,G,B> features) is given below (notice x,y vs. row,col!).

```
modelCovMatrix = [47.917 0 0 408.250

-146.636 68.487 -141.572 69.828 -123.269 53.479

-146.636 68.487 2654.285 2621.672 2440.381

-141.572 69.828 2621.672 2597.818 2435.368

-123.269; 53.479; 2440.381; 2435.368; 2404.923];
```

Test all possible 1-pixel overlapping windows (each of size 70 rows by 24 columns, with the upperleft-corner as the window origin) in the image with the given model. Save the match distance for each box location in the image at each pixel location (for the origin of the window). Plot/display the match-distance-image. Provide the **location of the best match** distance for the best candidate. Note that the above given covariance matrix is biased (normalized with 1/(M*N)), and Matlab's cov function is unbiased by default using 1/(M*N-1), so call `cov( X, 1 )` to make it consistent (biased). Leave the image with colors ranging 0-255 (do not scale/normalize the colors). **NOTE: make sure not to take a *log()* of zero at any time!** [5 pts]



Location1: x(column):255, y(row): 26



Location1: x(column):142, y(row): 34

The surface graph of the match-distance-image generated by the Matlab surf() function.

The match-distance-image generated by the Matlab imagesc() function.

The way I generate this matching distance matrix is by first computing the vectors for each window, and then computing the covariance matrix and eigenvalues. And then you sum the log eigenvalues squared, and then you put them in the matrix. And then you take the minimum in the matrix, and that gives you the third picture, and its position.

2) As usual, turn in and upload your material.

```
modelCovMatrix = [47.917 0 -146.636 -141.572 -123.269;
0 408.250 68.487 69.828 53.479;
-146.636 68.487 2654.285 2621.672 2440.381;
-141.572 69.828 2621.672 2597.818 2435.368;
-123.269 53.479 2440.381 2435.368 2404.923];

Im = double(imread('target.jpg'));

[y,x,z] = size(Im);
for r = 1: (y-69)
    for c = 1:(x-23)
        window(1:70,1:24,1:z) = Im(r:r+69,c:c+23,1:z);
        [j,i,k] = size(window);
        w = 1;
        for row = 1:j
            for column = 1:i
                window_fk(w,1:5) = [column,row,window(row,column,1),...
                    window(row,column,2),window(row,column,3)];
                w = w+1;
            end
        end
        candidatCovMatrix = cov(window_fk,1);
        eigenV = eig(modelCovMatrix,candidatCovMatrix);
        sum = 0;
        for e = 1:length(eigenV)
            if eigenV(e) ~= 0
                sum = sum + log(eigenV(e))^2;
            end
        end

       results(r,c) = sqrt(sum);

    end

end
surf(results);
imagesc(results);

result = min(results(:));
[r,c] = find(results==result);

disp(r)
disp(c)
result_Im(1:70,1:24,1:3) = Im(r:r+69,c:c+23,1:3);

imagesc(uint8(result_Im));
```

```
axis('image');
```