CORRELATION SIGNAL TO NON-STATIONARY NOISE RATIO PERFORMANCE

By Laurence G. Hassebrook

Updated with non-stationary noise section 3-28-2014

This visualization is used to demonstrate performance of image correlation in the presence of non-stationary colored noise

1. Input reference data is used to create a detection filter from: FSMC_Subject0000_0001_01.bmp. Note that this is scan 1 of finger 1 (i.e., right thumb)

![Reference Image](Figure 1: Reference Image)

2. A rectangular section is cropped from the image in Figure 1. This will be used as the filter. However, as is, it will respond to “dc” or constant intensity regions. To fix this, subtract its average value out so that its dc response will be zero. Specify size, location and show your crop region.
3. The test image is a different scan from the same subject and the same finger: FSPA_Subject0000_0002_01.bmp. Note that this is scan 2 of finger 1 (right thumb).
4. Zero pad the cropped region with zero values. The size of the final image is that of the test image. For matlab, we used the fftshift function to put the origin in the center. For many applications, you need to center the cropped section about the corner \{1,1\} and distribute in quadrants to the other corners. The zero padding is everything else. The reason that you want the cropped impulse response to be centered is so the correlation peak will occur at the center position of where the match occurs with the test image. Thus, with correlation you not only get detection but you also get location of the match.

The matlab code used for zero padding is:

```matlab
Asize=size(Abw);
Cbw=zeros(Asize(1),Asize(2));
Csize=size(Cbw)
Cbw( (1+(Csize(1)-Bsize(1))/2):((Csize(1)+Bsize(1))/2), (1+(Csize(2)-Bsize(2))/2):((Csize(2)+Bsize(2))/2))=Bbw(1:Bsize(1),1:Bsize(2));
```

Where the original image is \(\text{Abw}\) and the cropped section is \(\text{Bbw}\).

![Zero Padded Impulse Response](image)

**Figure 4:** Zero padded impulse response.
5. Correlation response from test image and detection filter is shown in Fig. 5. The white spot is the location of the match. Sometimes this is visualized by using “vga” like colars, marking the peak with cross hairs and/or using a dB based plot. Also, a negative image can be used to visualize a response that is predominantly black which also saves ink in printing out the results.

The matlab code for the correlation is:

```matlab
% correlation
FCbw=conj(fft2(Cbw));
FAbw=fft2(Abw);
FCcorr=FAbw.*FCbw;
corr=abs(fftshift(ifft2(FCcorr)));
```

where Abw is the full test image and Cbw is the zero padded partition.

![Negative Correlation Intensity](image)

Figure 5: Negative of Correlation Response.

6. The correlation is commonly viewed in 3-Dimensions as shown in Fig. 6. The matlab code for viewing 3-D surfaces is:

```matlab
surf(corr);
shading interp
```
7. Perform the steps 1 through 6 but with a non-stationary colored noise image added to a 3x3 augmentation of the test image. Adjust the size of the crop so that at the worst response, the peak is still visible when displayed as in Fig. 6. The reason we present this approach is not strictly for analysis but more for presentation and visualization. By augmenting the test prints and then adding them to non-stationary noise followed by correlation with the desired partition, we can see and compare the detection performance within one image thereby conveying the range of performance with a single figure rather than 9 figures.

To generate the non-stationary noise we start with a Gaussian noise image A such that:

\[ Nx = 3 \times \text{Asize}(2); \]
\[ My = 3 \times \text{Asize}(1); \]
\[ A = \text{randn}(My, Nx); \]

Where Asize is the original size of the test image. Hence, the noise image will be 3 times the size along both the horizontal and vertical directions.
Another weighting image that varies from left to right in a linear gradient is created. We will call this image B and is created in MATLAB with:

```matlab
B = ones(My, Nx);
for m = 1:My
    for n = 1:Nx
        B(m, n) = (n-1) / (Nx-1);
    end;
end;
```
If we multiply gradient image B time noise image A, then we have a noise image whose standard deviation varies from left to right as 0 to 1. The result of A.*B is shown in Fig. 7.
So the image in Fig. 7 is non-stationary in variance from left to right. We now want to introduce a variation in coloring of the noise from top to bottom. The matlab code for this is tricky because we will vary a rectangular kernel’s area as it is convolved from top to bottom. We actually start with the original matrix A again and perform the coloring operation. The MATLAB code we used for this is:

```matlab
% Vertically Non-stationary Noise coloring
Z=zeros(My,Nx);
for m=1:My
    for n=1:Nx
        M=floor((m*5)/(Nx-1)); % Averaging window size
        cm=1/(2*M+1); % Normalization coefficient
        for i=-M:M
            for j=-M:M
                u=floor(m+i);
                v=floor(n+j);
                if (u>0) & (u<=My) & (v>0) & (v<=Nx)
                    Z(m,n)=Z(m,n)+A(u,v)*cm;
                end;
            end; % j
        end; % i
    end; % n
end; % m
```
The last step is to use the gradient image on the colored noise image as $Y = Z \ast B$ in MATLAB.
Figure 9: The final non-stationary noise image.

The augmentation of the fingerprints is accomplished with the MATLAB code and then is normalized to have a peak of 1. This way the experiment can be repeated with any type of data and the resulting SNR will be similar. The code for this is:

```matlab
Abw3x3=[Abw,Abw,Abw;Abw,Abw,Abw;Abw,Abw,Abw];
% normalize peak to 1
Abw3x3=Abw3x3/max(max(Abw3x3));
```
Figure 10: Augmented copies of the test print.

The augmented fingerprints are shown in Fig. 10.
The test print matrix and the non-stationary noise are added and shown in Fig. 11. Just as we did before with a single test print, we form zero padding for the full 3x3 image augmentation as:

```
% zero pad Filter
Asize3x3 = size(Abw3x3);
Cbw = zeros(Asize3x3(1), Asize3x3(2));
Csize3x3 = size(Cbw);
Cbw((1 + (Csize3x3(1) - Bsize(1))/2):((Csize3x3(1) + Bsize(1))/2), (1 + (Csize3x3(2) - Bsize(2))/2):((Csize3x3(2) + Bsize(2))/2)) = Bbw(1:Bsize(1), 1:Bsize(2));
```
We perform the correlation as before such that:

```matlab
% correlation
FCbw=conj(fft2(Cbw));
FBbw=fft2(Bbw3x3);
FCcorr=FBbw.*FCbw;
corr=abs(fftshift(ifft2(FCcorr)));
```

The problem now is that MATLAB will not show the peaks because of the method it uses to downsample data displayed in the `imagesc` function. This brings up the need for a technique commonly used in correlation display of downsampled data with preservation of the maximum values. In this case, we tiled the 3000x3000 image with 300 by 300 tiles. Then we find the maximum value in each tile and plot the 300x300 result. This works well for detection analysis because we not only want to know the maximum peak detection but also the maximum sidelobe values. We then add another layer of clarification by binarizing the result to simplify the display of the peak and sidelobe values above a threshold value. The code for this is:

```matlab
% downsample corr but keep local maximums
Ndec1=10;
Nxd=floor(Asize3x3(2)/Ndec1);
Myd=floor(Asize3x3(1)/Ndec1);
```
sized = [Myd, Nxd];
corrd = zeros (Myd, Nxd);
for n=1:Nxd
    for m=1:Myd
        md = 1+ (m-1)*Ndeci;
        nd = 1+ (n-1)*Ndeci;
        partition = corr (md: (md+Ndeci), nd: (nd+Ndeci));
        corrd (m,n) = max (max (partition));
    end;
end;
% corrd = corr;
corrbinary = corrd;
% binarized correlation
thresh = 0.7
J = find (corrd < (thresh * max (max (corrd))));
corrbinary (J) = 0;
K = find (corrd >= (thresh * max (max (corrd))));
corrbinary (K) = 1;

Figure 13: Binarized and thresholded negative correlation response.
APPENDIX: MATLAB CODE

1. Correlation code: Cbw is the correlation filter impulse response and Abw is the test image.

   \[
   \text{FCbw} = \text{conj}(\text{fft2}(\text{Cbw})) ; \\
   \text{FAbw} = \text{fft2}(\text{Abw}) ; \\
   \text{FCcorr} = \text{FAbw} \cdot \text{FCbw} ; \\
   \text{corr} = \text{abs}(\text{ifftshift}(\text{ifft2}(\text{FCcorr}))) ;
   \]

2. Code for visualizing correlation in 3-D in MATLAB. The correlation is real only and stored in “corr”.

   \[
   \text{figure (5)} \\
   \%\text{colormap(gray);} \\
   \text{colormap(vga);} \\
   \text{surf(corr);} \\
   \]
shading interp
title('3-D surface topology of correlation')
print -djpeg figure6

3. Augmentation

B=[A,A,A;A,A;A,A,A]