Charles Hodgins
EECE566 - Detection Theory
Professor Jessica Fridrich
May 11, 2025

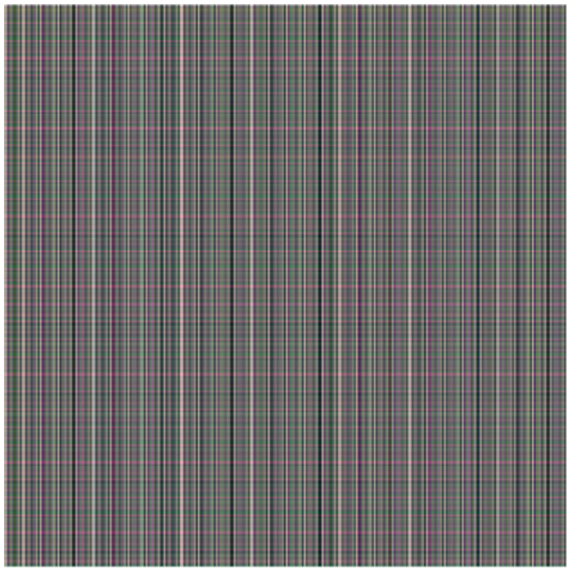
Sensor Fingerprint Digital Forensics Project

This project involves applying detection theory techniques to identify images that originate from the same camera sensor. This is done by utilizing the photo response non-uniformity (PRNU) signal calculated by processing images that come from a unique camera sensor. The signal is processed alongside a dataset of images containing 2 images from the same camera sensor, and other images from other camera sensors.

To begin the sensor fingerprint project, I cleaned the PRNU signal from the Canon 6D camera by removing the row and column means from the signal, which otherwise would not necessarily be unique to the Canon 6D fingerprint. Using my function preprocess_prnu
This calculates the column and row means and subtracts it from the signal to clean the fingerprint of non-unique noise.

```
function K_processed = preprocess_prnu(K)
   K_processed = K;
   for c = 1:3
        col_means = mean(K_processed(:,:,c), 1);
        K_processed(:,:,c) = K_processed(:,:,c) - col_means;
        row_means = mean(K_processed(:,:,c), 2);
        K_processed(:,:,c) = K_processed(:,:,c) - row_means;
   end
end
```

Calculating the difference between the two PRNU matrices shows just how much noise was removed from the camera, shown in a **256x256 cropping of the matrix**, resulting in a pattern of column and row averages:



After the K.mat PRNU is cleaned, I filter the noise residual out of each channel in every image individually. This noise residual is returned to the value **W** which will be used to cross correlate with the PRNU signal and eventually calculate the pce score.

```
function W = get_noise_residual(img)
   W = double(zeros(size(img)));
   for c = 1:3
       channel = img(:,:,c);
       W(:,:,c) = channel - wiener2(channel);
   end
   W = 0.3*W(:,:,1) + 0.6*W(:,:,2) + 0.1*W(:,:,3);
end
```

The next step is to calculate the **peak to correlation energy** following this formula:

$$\mathrm{PCE} = \frac{\mathbf{NCC}(u_{peak}, v_{peak})^2}{\frac{1}{mn - |\Lambda|} \sum_{(u,v)\notin\Lambda} \mathbf{NCC}(u,v)^2}$$

normalized detection statistic, easier to set threshold

With the matlab function:

```
function [pce, ncc] = compute_pce(W, K, exclude_region)

   pad_rows = size(K,1) - size(W,1);
   pad_cols = size(K,2) - size(W,2);

   if pad_rows > 0 || pad_cols > 0
       W = padarray(W, [max(0,pad_rows), max(0,pad_cols)], 0, 'post');
   else
       W = W(1:size(K,1), 1:size(K,2));
   end
```

```
    ncc = crosscorr2(W, K);

    [peak_vals, peak_rows] = max(ncc);
    [peak_val, peak_col] = max(peak_vals);
    peak_row = peak_rows(peak_col);

    [m, n] = size(ncc);
    [X, Y] = meshgrid(1:n, 1:m);
    exclusion_mask = (abs(X-peak_col) <= exclude_region/2) & ...
                     (abs(Y-peak_row) <= exclude_region/2);

    ncc_squared = ncc.^2;
    ncc_squared(exclusion_mask) = 0;
    energy = sum(ncc_squared(:)) / (m*n - sum(exclusion_mask(:)));

    pce = peak_val^2 / (energy + eps);
end
```

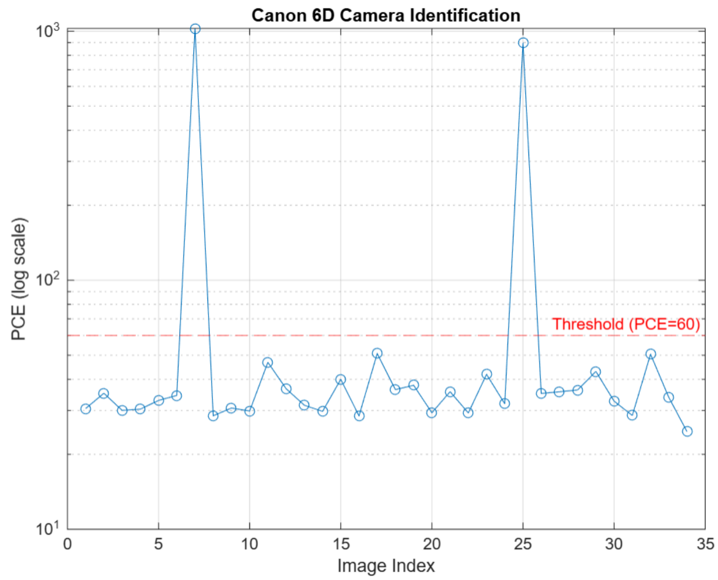
It is crucial to remove the section of the 5x5 area around the peak value, the exclusion mask, from the ncc when calculating the squared mean within the denominator of the pce equation, as well as to make sure the smaller signal is padded with zeros, instead of cropping the larger signal to make sure that the dimensions match when calculating. These were initial errors I ran into when creating the pce calculation function.

This function utilizes the crosscorr2.m function which calculates the cross correlation using the fourier transform to simplify the calculation into a multiplication in the fourier domain. The camera_identification() function acts as a main function to call the rest and perform necessary loading and figure generation.

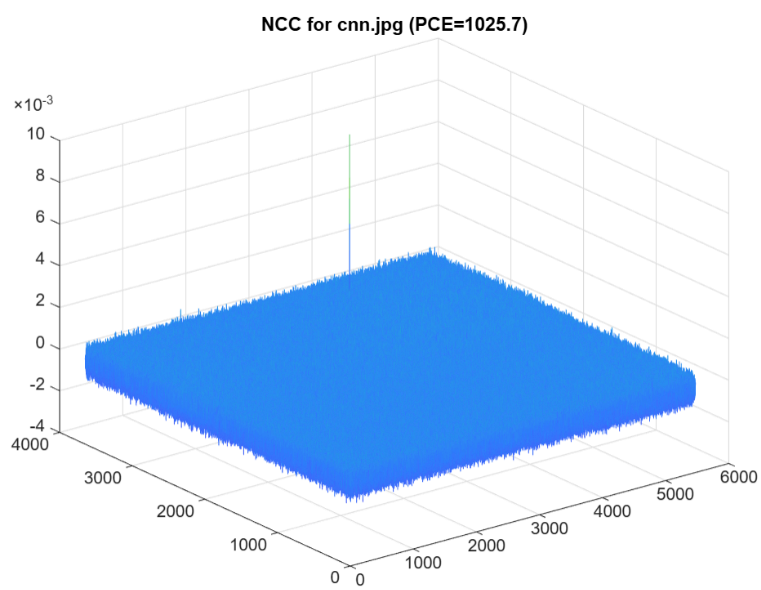
```
function camera_identification()
    try
        load('K.mat', 'K');
        K_processed = preprocess_prnu(K);
        figure(1);
        K_gray = 0.3*K_processed(:,:,1) + 0.6*K_processed(:,:,2) +
0.1*K_processed(:,:,3);
        imshow(mat2gray(K(1:256,1:256,:)-K_processed(1:256,1:256,:)))
    catch ME
        fprintf('Error loading PRNU: %s\n', ME.message);
        return;
    end
    image_dir = 'All_34_images';
    image_files = dir(fullfile(image_dir, '*.jpg'));
    pce_results = cell(length(image_files), 2);

    for i = 1:length(image_files)
        try
```

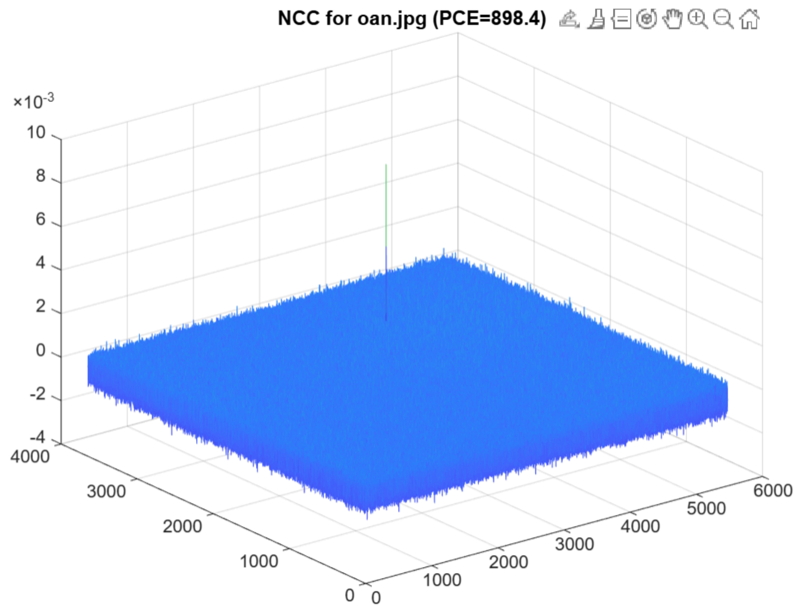
```
            img_path = fullfile(image_dir, image_files(i).name);
            fprintf('Processing %s...\n', image_files(i).name);
            img = double(imread(img_path)) / 255.0;
            if ~validate_conversion(img)
                continue;
            end
            W = get_noise_residual(img);
            [pce, ncc] = compute_pce(W, K_gray, 5);
            pce_results{i,1} = image_files(i).name;
            pce_results{i,2} = pce;
            if pce > 60
                figure(i);
                subplot(1,2,1);
                mesh(ncc);
                title(sprintf('NCC for %s (PCE=%.1f)', image_files(i).name,
pce));
                subplot(1,2,2);
                imshow(img);
                title('Original Image');
            end
        catch ME
            fprintf('Error processing %s: %s\n', image_files(i).name,
ME.message);
            continue;
        end
    end
    fprintf('\nCamera Identification Results:\n');
    [~, idx] = sort(cell2mat(pce_results(:,2)), 'descend');
    for i = 1:length(idx)
        fprintf('%s: %.1f\n', pce_results{idx(i),1}, pce_results{idx(i),2});
    end
    figure(2);
    semilogy(cell2mat(pce_results(:,2)), 'o-');
    hold on;
    yline(60, 'r--', 'Threshold (PCE=60)');
    xlabel('Image Index');
    ylabel('PCE (log scale)');
    title('Canon 6D Camera Identification');
    grid on;
end
```

Each image is loaded, the corresponding pce is calculated upon the images noise residual and the K_processed PRNU of the Canon 6D sensor fingerprint. The pce scores that are significantly above the threshold of about 60, are confidently the images belonging to the same sensor. The figure below identifies all of the pce scores for the 34 images, with two values reaching towards 10^3 while the rest of them score just below the threshold.



The next two figures are the corresponding mesh graphs of the normalized cross correlation matrices, identifying the names of the images as **'cnn.jpg'** and **'oan.jpg'** as the two images that belong to the same PRNU given as 'K.mat'. The mesh plots show a smooth surface on the ncc matrix except for an exceptionally high peak value that indicates that W and K match. The location of the peak value in the ncc also determines the cropping position of the image relative to the camera. The shift in the peak confirms that the image was cropped and shifted from the center because the peak is not in the center of the ncc plot for either image.





For reference, below is the same mesh plot of the ncc for an image that did not correlate well with the PRNU, and scored a low pce.