

Contents

- [LED reflection cleanser](#)
- [read the data from disk](#)
- [plot what happens at a few pixels](#)
- [time-series plotting](#)
- [histogram analysis](#)
- [identify pixels affected by most prominent reflection](#)
- [take out LED reflections](#)
- [cleanup](#)

```
function data = LEDcleaner(varargin)
```

LED reflection cleanser

A program that takes out the flashing LED reflection from the Jan 13, 2013 DMC CCD data. A nice mutual observation between CMOS and CCD, but CCD has really bright LED flashing. We want to see if there is a straightforward way to remove the LED from this data.

A primitive mathematical model for the "bad" frames is: $Y = X + e$

where:

Y is the corrupted image

X is the original image

e is the 2D distribution of time-varying LED reflections

One approach might be to estimate a time-varying 2D \hat{e} , and find an estimate \hat{X} of the original uncorrupted image by $\hat{X} = Y - \hat{e}$.

As a first observation, we see that e is relatively spatially constant --there are a couple flickers to other parts of the image, but once we figure out how to get rid of one reflection, we can get rid of them all.

We further observe that e is nearly binary with time -- there is still some time-varying component within the "on" times, but it appears there is some spatial correlation of the LED on-intensity variance that might be exploited to enhance LED reflection removal

Complicating our LED reflection estimate \hat{e} is that the image X is riddled with time-varying noise.

```
addpath('histutils')
```

```
p = inputParser;
addParamValue(p,'datadir','~/data')
addParamValue(p,'writevid',[]) %#ok<*NVREPL> % need this for Octave 4.0 which doesn't have
```

```

addParameter
parse(p,varargin{:})
U = p.Results;

BigFN = [U.datadir,filesep,'2013-01-13T21-00-00_frames_562000-1-568000.DMCdata'];

```

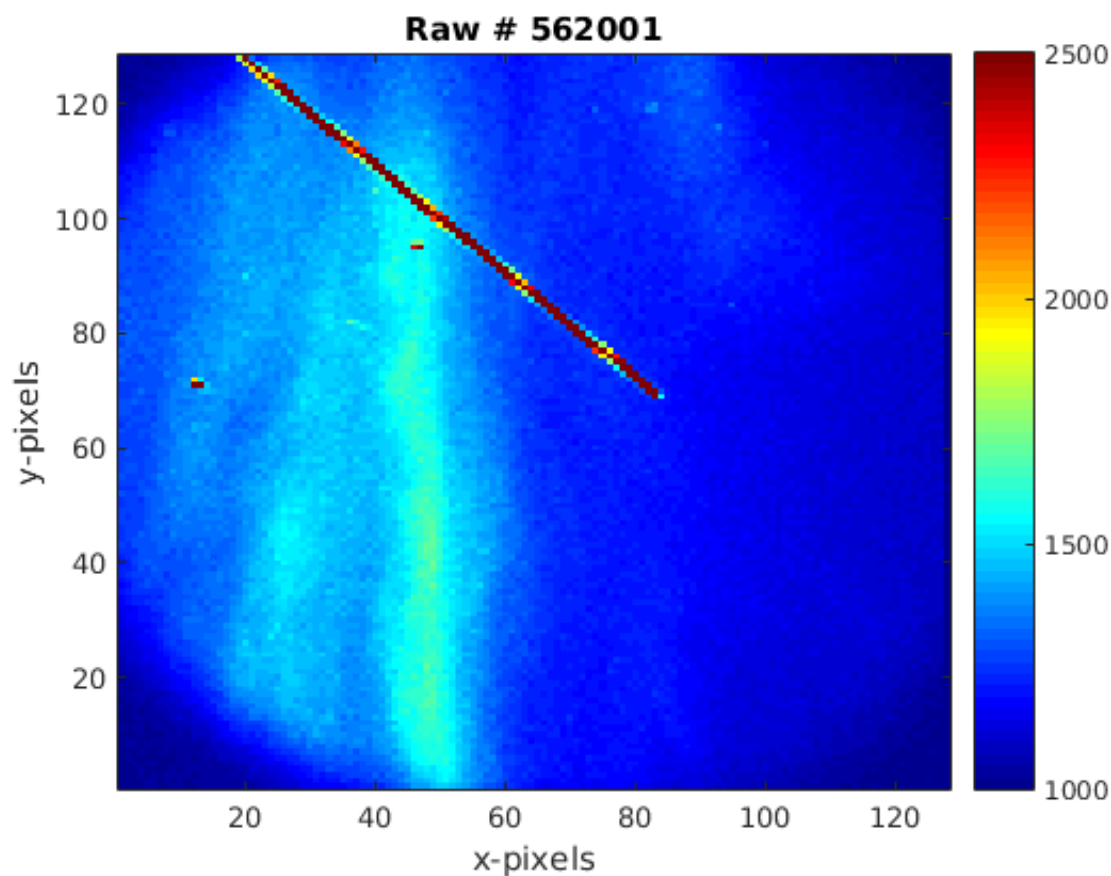
read the data from disk

```

[data,rawFrameInd] = rawDMCreader(BigFN,512,512,4,4,'all',0,[1000,4000]);
nFrame = length(rawFrameInd);
% plot example "bad" frame with LED reflection
badFrame = double(data(:,:,2));

figure(10),clf(10),set(10,'name','image with LED reflection')
ax10=axes('parent',10);
imagesc(badFrame,[1000 2500]);
set(ax10,'ydir','normal')
colormap('jet')
xlabel(ax10,'x-pixels'), ylabel(ax10,'y-pixels')
title(ax10,['Raw # ',int2str(rawFrameInd(2))])
colorbar('peer',ax10)

```



plot what happens at a few pixels

pick a couple pixels to plot over time

```

ySel = [69, 128,68,127,126];
xSel = [83, 20, 83,20, 20 ];

% hist() requires single or double data; Octave requires single or double
% for line() and plot()

% bad pixels
pixData(:,1) = double(data(ySel(1),xSel(1),:));
pixData(:,2) = double(data(ySel(2),xSel(2),:));
% neighboring pixels to the bad pixels
pixData(:,3) = double(data(ySel(3),xSel(3),:));
pixData(:,4) = double(data(ySel(4),xSel(4),:)); %still bad!
pixData(:,5) = double(data(ySel(5),xSel(5),:)); %better

```

time-series plotting

first plot over time, then we'll zoom in for some clues... observe the spatial LED correlation within the "on" times from widely spatially separated pixels

```

figure(2),clf(2),set(2,'name','Time-series')

subplot(2,1,1)
line(1:nFrame, pixData(:,1) , 'color','r')
line(1:nFrame, pixData(:,3) , 'color','b')
ylabel('Data Numbers')
axis('tight')
legend('bad pixel','neighbor pixel','location','best')
title(['(x,y) =(',int2str(xSel([1,3])),',',int2str(ySel([1,3])),')'])

subplot(2,1,2)
line(1:nFrame, pixData(:,2) , 'color','r')
line(1:nFrame, pixData(:,4) , 'color','m')
line(1:nFrame, pixData(:,5) , 'color','b')
% label the axes
ylabel('Data Numbers')
xlabel('File Frame # ')
title(['(x,y) =(',int2str(xSel([2,4,5])),',',int2str(ySel([2,4,5])),')'])
axis('tight')

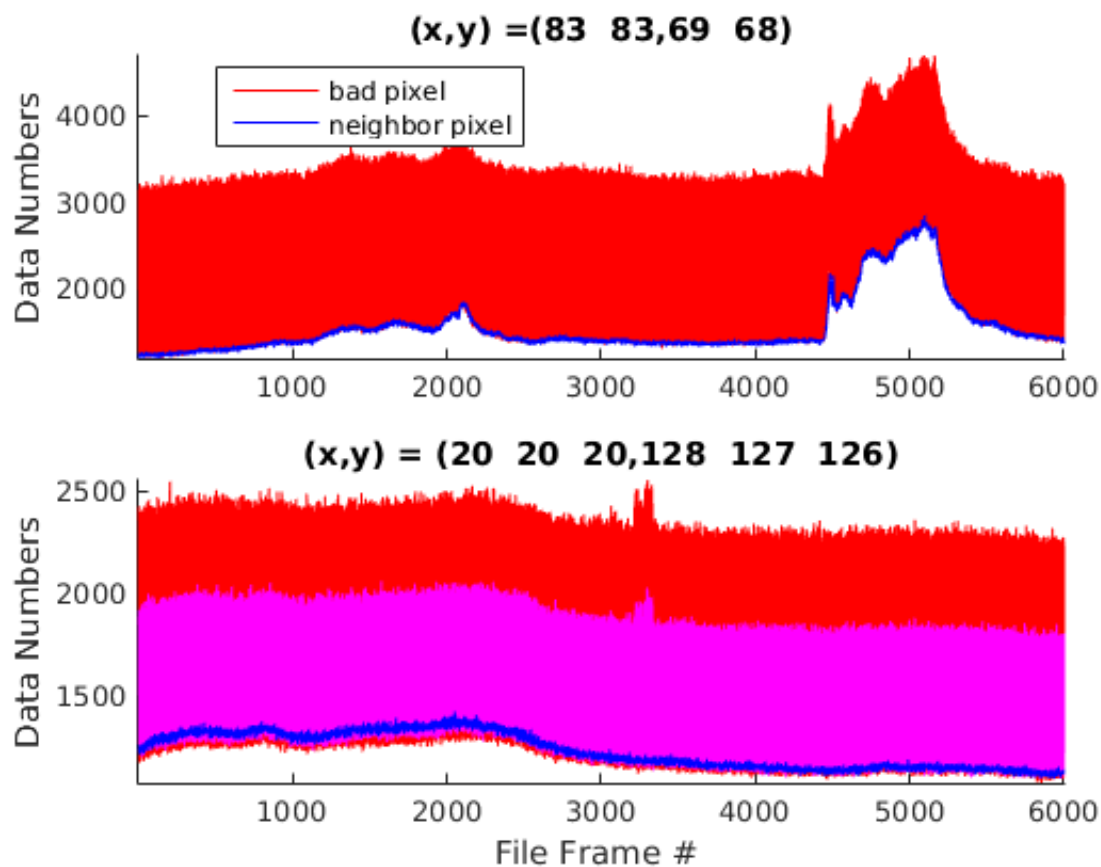
% now the zoomed-in version
zoomTimeInd = 1:200;
figure(12),clf(12),set(12,'name','Zoomed Time-series')

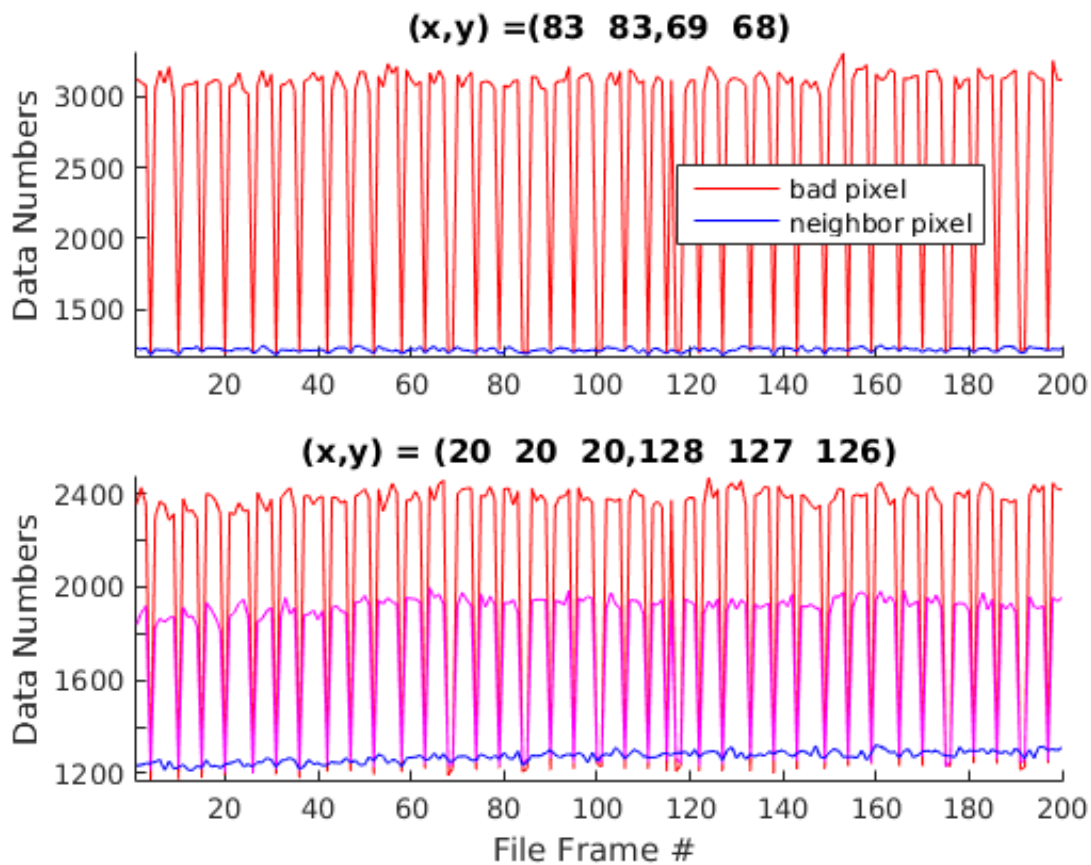
subplot(2,1,1)
line(zoomTimeInd, pixData(zoomTimeInd,1) , 'color','r')
line(zoomTimeInd, pixData(zoomTimeInd,3) , 'color','b')
ylabel('Data Numbers')
axis('tight')
legend('bad pixel','neighbor pixel','location','best')
title(['(x,y) =(',int2str(xSel([1,3])),',',int2str(ySel([1,3])),')'])

subplot(2,1,2)
line(zoomTimeInd, pixData(zoomTimeInd,2) , 'color','r')
line(zoomTimeInd, pixData(zoomTimeInd,4) , 'color','m')
line(zoomTimeInd, pixData(zoomTimeInd,5) , 'color','b')

```

```
% label the axes  
ylabel('Data Numbers')  
xlabel('File Frame # ')  
title(['(x,y) = (',int2str(xSel([2,4,5])),',',int2str(ySel([2,4,5])),',')'])  
axis('tight')
```



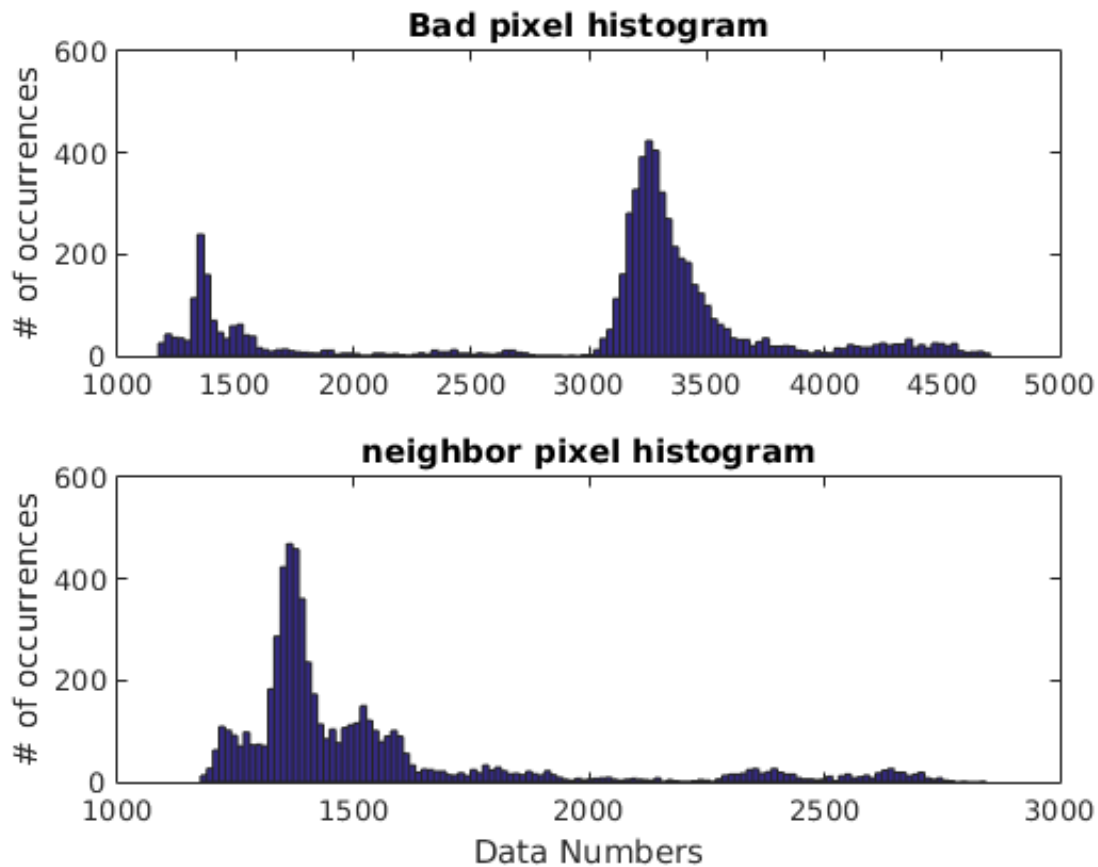


histogram analysis

histograms tell us about the pdf (probability density function) of data. How likely certain values are vs. others.

```
figure(3),clf(3)
subplot(2,1,1)
hist( pixData(:,1),128)
%set(gca,'yscale','log')
ylabel('# of occurrences')
title('Bad pixel histogram')

subplot(2,1,2)
hist( pixData(:,3), 128)
ylabel('# of occurrences')
xlabel('Data Numbers')
title('neighbor pixel histogram')
```



identify pixels affected by most prominent reflection

Most prominent reflection starts at lower right and heads toward/past image center. I.e. pixels $(x,y) = (128,20)$ to $(69,83)$

hmm, what does the spatial derivative look like? have to find 2D gradient as vertical and horizontal spatial derivatives

```

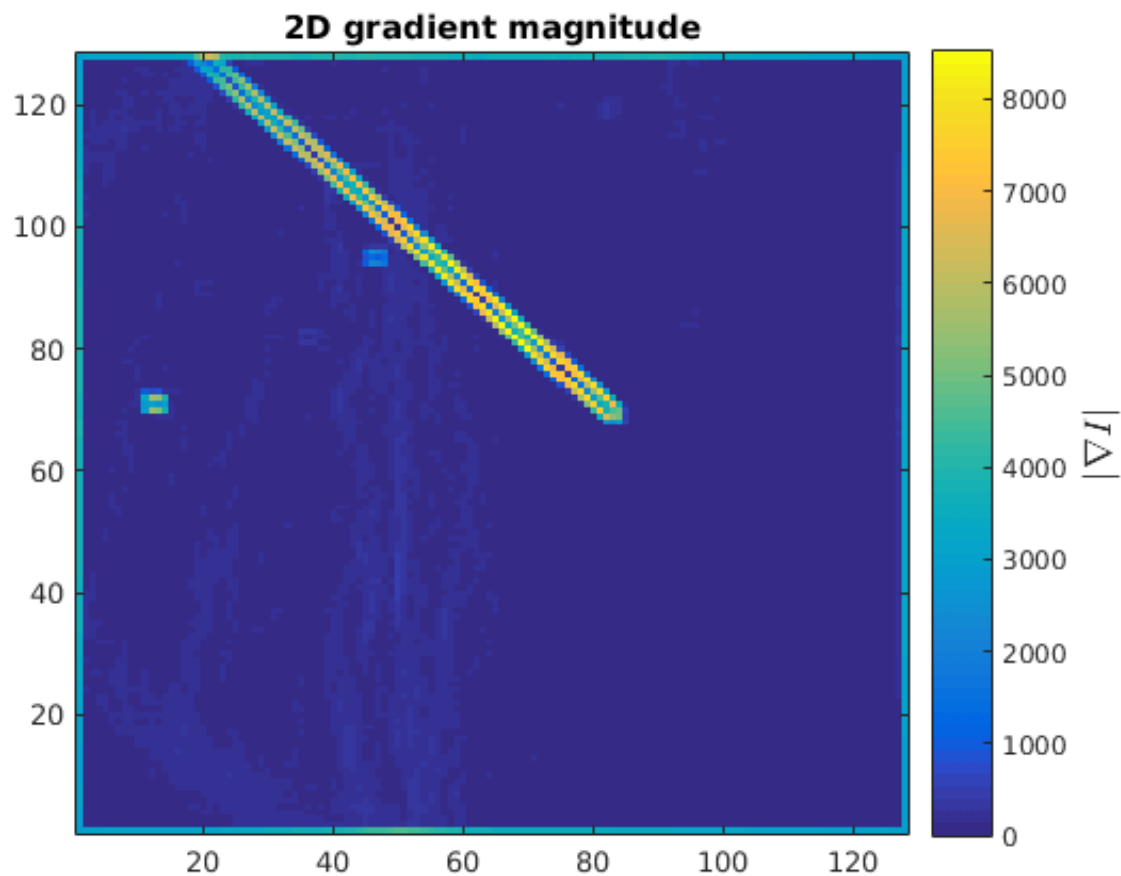
horizKern = fspecial('prewitt'); %from image processing theory
vertKern = transpose(horizKern);

horizGradient = conv2(badFrame,horizKern,'same');
vertGradient = conv2(badFrame,vertKern,'same');

gradientMagnitude = hypot(horizGradient,vertGradient);

figure(11),clf(11)
imagesc(gradientMagnitude)
set(gca,'ydir','normal')
hcb = colorbar;
ylabel(hcb,' $|\nabla I|$ ','interpreter','latex','fontsize',14)
title('2D gradient magnitude')

```



take out LED reflections

subtract the reflection from impacted frames only ??? $\hat{X} = Y - \hat{e}$

assign a 2D \hat{e} mask.

cleanup

```
if ~nargout, clear, end %stops lots of junk text from filling up console window
```

```
end %function
```

Published with MATLAB® R2015a