Digital Image Processing Notes

MATLAB Plots of LPF and HPF

Filtering:

TBD performance of matlab vs. GPU
http://www.advancedsourcecode.com/ffw.asp

Convolution Thm:

Assuming Linear Spatially invariant system a system is characterized by it's impulse response. For the cases of images which we denote with a 2dim grey scale matrix, we define the impulse response to be a function of the number of rows and columns in the matrix h(n1,n2)

The output of the LSI is the convolution of the input and the impulse response

y(n1,n2)=x(n1,n2)**h(n1,n2)

To find an impulse response of a system input a delta function and the output is the impulse response. The equivalent of a 2d delta function for images is the matrix with a 1 at 0,0 and zeros everywhere else.

0 0 0
0 1 0
0 0 0

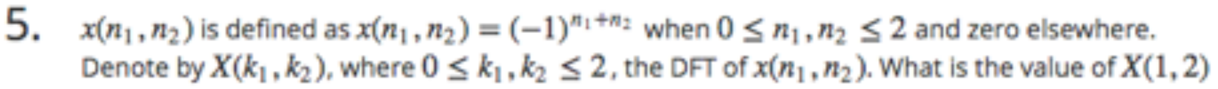
\begin{pmatrix}
0 &0 &0 \\
0 &1 &0 \\
0 &0 &0
\end{pmatrix}

Edge Effects Convolution

Week 3 Questions:

A FFT is a way to compute the DFT. There are butterfly algorithms to calculate the FFT fast; this is different than what the GPU can do.

Find DFT of (-1)^(n1+n2)



5. $x(n_1, n_2)$ is defined as $x(n_1, n_2) = (-1)^{n_1+n_2}$ when $0 \leq n_1, n_2 \leq 2$ and zero elsewhere. Denote by $X(k_1, k_2)$, where $0 \leq k_1, k_2 \leq 2$, the DFT of $x(n_1, n_2)$. What is the value of $X(1, 2)$

Defn of FFT/DFT:

X(k1,k2)=sum(sum(x(n1,n2)exp(-j2pi/N1*k1)exp(-j2pi/N2*k2)))

X(k1,k2)=sum(n1 from 0 to N1-1)(sum(n2 from 0 to N2-1)(x(n1,n2)exp(-j2pi/N1*n1*k1)exp(-j2pi/N2*n2*k2)))

X(k1,k2)=
x(0,0)exp(-j2pi/N1*0*k1)exp(-j2pi/N2*0*k2)+
x(0,1)exp(-j2pi/N1*0*k1)exp(-j2pi/N2*1*k2)+
x(0,2)exp(-j2pi/N1*0*k1)exp(-j2pi/N2*2*k2)+
x(1,0)exp(-j2pi/N1*1*k1)exp(-j2pi/N2*0*k2)+
x(1,1)exp(-j2pi/N1*1*k1)exp(-j2pi/N2*1*k2)+
x(1,2)exp(-j2pi/N1*1*k1)exp(-j2pi/N2*2*k2)+
x(2,0)exp(-j2pi/N1*2*k1)exp(-j2pi/N2*0*k2)+
x(2,1)exp(-j2pi/N1*2*k1)exp(-j2pi/N2*1*k2)+
x(2,2)exp(-j2pi/N1*2*k1)exp(-j2pi/N2*2*k2)

Set the exp terms with 0 to 1

X(k1,k2)=
x(0,0)+
x(0,1)exp(-j2pi/N2*1*k2)+
x(0,2)exp(-j2pi/N2*2*k2)+
x(1,0)exp(-j2pi/N1*1*k1)+
x(1,1)exp(-j2pi/N1*1*k1)exp(-j2pi/N2*1*k2)+
x(1,2)exp(-j2pi/N1*1*k1)exp(-j2pi/N2*2*k2)+
x(2,0)exp(-j2pi/N1*2*k1)+
x(2,1)exp(-j2pi/N1*2*k1)exp(-j2pi/N2*1*k2)+
x(2,2)exp(-j2pi/N1*2*k1)exp(-j2pi/N2*2*k2)

X(1,2)=
x(0,0)+
x(0,1)exp(-j2pi/N2*2)+
x(0,2)exp(-j2pi/N2*4)+
x(1,0)exp(-j2pi/N1)+
x(1,1)exp(-j2pi/N1)exp(-j2pi/N2*2)+
x(1,2)exp(-j2pi/N1)exp(-j2pi/N2*4)+
x(2,0)exp(-j2pi/N1*2)+
x(2,1)exp(-j2pi/N1*2)exp(-j2pi/N2*2)+
x(2,2)exp(-j2pi/N1*2)exp(-j2pi/N2*4)

x(0,0)=(-1)^0=1
x(1,0)=x(0,1)=(-1)^1=-1
x(1,2)=x(2,1)=(-1)^3=-1
x(2,2)=(-1)^4=1

X(1,2)=
1
-exp(-j2pi/N2*2)
+exp(-j2pi/N2*4)
-exp(-j2pi/N1)
+exp(-j2pi/N1)exp(-j2pi/N2*2)
-exp(-j2pi/N1)exp(-j2pi/N2*4)
+exp(-j2pi/N1*2)
-exp(-j2pi/N1*2)exp(-j2pi/N2*2)
+exp(-j2pi/N1*2)exp(-j2pi/N2*4)

N2=N1=3

X(1,2)=
1
-exp(-j2pi/3*2)
+exp(-j2pi/3*4)
-exp(-j2pi/3)
+exp(-j2pi/3)exp(-j2pi/3*2)
-exp(-j2pi/3)exp(-j2pi/3*4)
+exp(-j2pi/3*2)
-exp(-j2pi/3*2)exp(-j2pi/3*2)
+exp(-j2pi/3*2)exp(-j2pi/3*4)

combine constants in exp

X(1,2)=
1
-exp(-j4pi/3)
+exp(-j8pi/3)
-exp(-j2pi/3)
+exp(-j2pi/3)exp(-j4pi/3)
-exp(-j2pi/3)exp(-j8pi/3)
+exp(-j4pi/3)
-exp(-j4pi/3)exp(-j4pi/3)
+exp(-j4pi/3)exp(-j8pi/3)

Combine exp terms into single exp term

X(1,2)=
1
-exp(-j4pi/3)
+exp(-j8pi/3)

-exp(-j2pi/3)
+exp(-j6pi/3)
-exp(-j10pi/3)
+exp(-j4pi/3)
-exp(-j8pi/3)
+exp(-j12pi/3)

X(1,2)=
1
-exp(-j4pi/3)
+exp(-j8pi/3)
-exp(-j2pi/3)
+exp(-j6pi/3)
-exp(-j10pi/3)
+exp(-j4pi/3)
-exp(-j8pi/3)
+exp(-j12pi/3)

e^npi = cos(npi) + isin(npi)

X(1,2)=1-exp(-j4pi/3)+exp(-j8pi/3)-exp(-j2pi/3)+exp(-j2pi)-exp(-j10pi/3)+exp(-j4pi/3)-exp(-j8pi/3)+exp(-j4pi)

1
-(cos(-4pi/3)+jsin(-4pi/3))
+(cos(-8pi/3)+jsin(-8pi/3))
-(cos(-2pi/3)+jsin(-2pi/3))
+(cos(-2pi)+jsin(-2pi))
-(cos(-10pi/3)+jsin(-10pi/3))
+(cos(-4pi/3)+jsin(-4pi/3))
-(cos(-8pi/3)+jsin(-8pi/3))
+(cos(-4pi)+jsin(-4pi))

distribute - signs

1-cos(-4pi/3)-jsin(-4pi/3)+cos(-8pi/3)+jsin(-8pi/3)-cos(-2pi/3)-jsin(-2pi/3)+cos(-2pi)+jsin(-2pi)-cos(-10pi/3)-jsin(-10pi/3)+cos(-4pi/3)+jsin(-4pi/3)-cos(-8pi/3)-jsin(-8pi/3)+cos(-4pi)+jsin(-4pi)

combine cos and sin terms

1-cos(-4pi/3)+cos(-8pi/3)-cos(-2pi/3)+cos(-2pi)-cos(-10pi/3)+cos(-4pi/3)-cos(-8pi/3)+cos(-4pi)-jsin(-4pi/3)+jsin(-8pi/3)-jsin(-2pi/3)+jsin(-2pi)-jsin(-10pi/3)+jsin(-4pi/3)-jsin(-8pi/3)+jsin(-4pi)

1
-cos(-4pi/3)
+cos(-8pi/3)

-cos(-2pi/3)
+cos(-2pi)
-cos(-10pi/3)
+cos(-4pi/3)
-cos(-8pi/3)
+cos(-4pi)
-jsin(-4pi/3)
+jsin(-8pi/3)
-jsin(-2pi/3)
+jsin(-2pi)
-jsin(-10pi/3)
+jsin(-4pi/3)
-jsin(-8pi/3)
+jsin(-4pi)

cos(-4pi/3) = -cos(-pi/3)=cos(pi/3)
cos(-8pi/3) = cos(-2pi/3)=cos(2pi/3)
cos(-2pi/3) = cos(2pi/3)
cos(-2pi)= 1
cos(-10pi/3)=cos(-4pi/3)=-cos(pi/3)
cos(-4pi)=1

sin(-4pi/3)= sin(pi/3)
sin(-8pi/3)=sin(-2pi/3)=-sin(2pi/3)
sin(-2pi/3)= - sin(2pi/3)
sin(-2pi)=0
sin(-10pi/3)=sin(-4pi/3)= sin(pi/3)
sin(-4pi)=0

1
-cos(pi/3)
+cos(2pi/3)
-cos(2pi/3)
+1
+cos(pi/3)
+cos(pi/3)
-cos(2pi/3)
+1
-jsin(pi/3)
-jsin(2pi/3)
+jsin(2pi/3)
-jsin(pi/3)
+jsin(pi/3)
+jsin(2pi/3)

cancel terms

1+1+cos(pi/3)-cos(2pi/3)+1-jsin(pi/3)+jsin(2pi/3)

3+1/2+1/2+1=4

DFT example:

X(k1,k2)=sum(sum(x(n1,n2)exp(-j2pi/N1*k1)exp(-j2pi/N2*k2)))

arithmetic sum 16*17/2=136

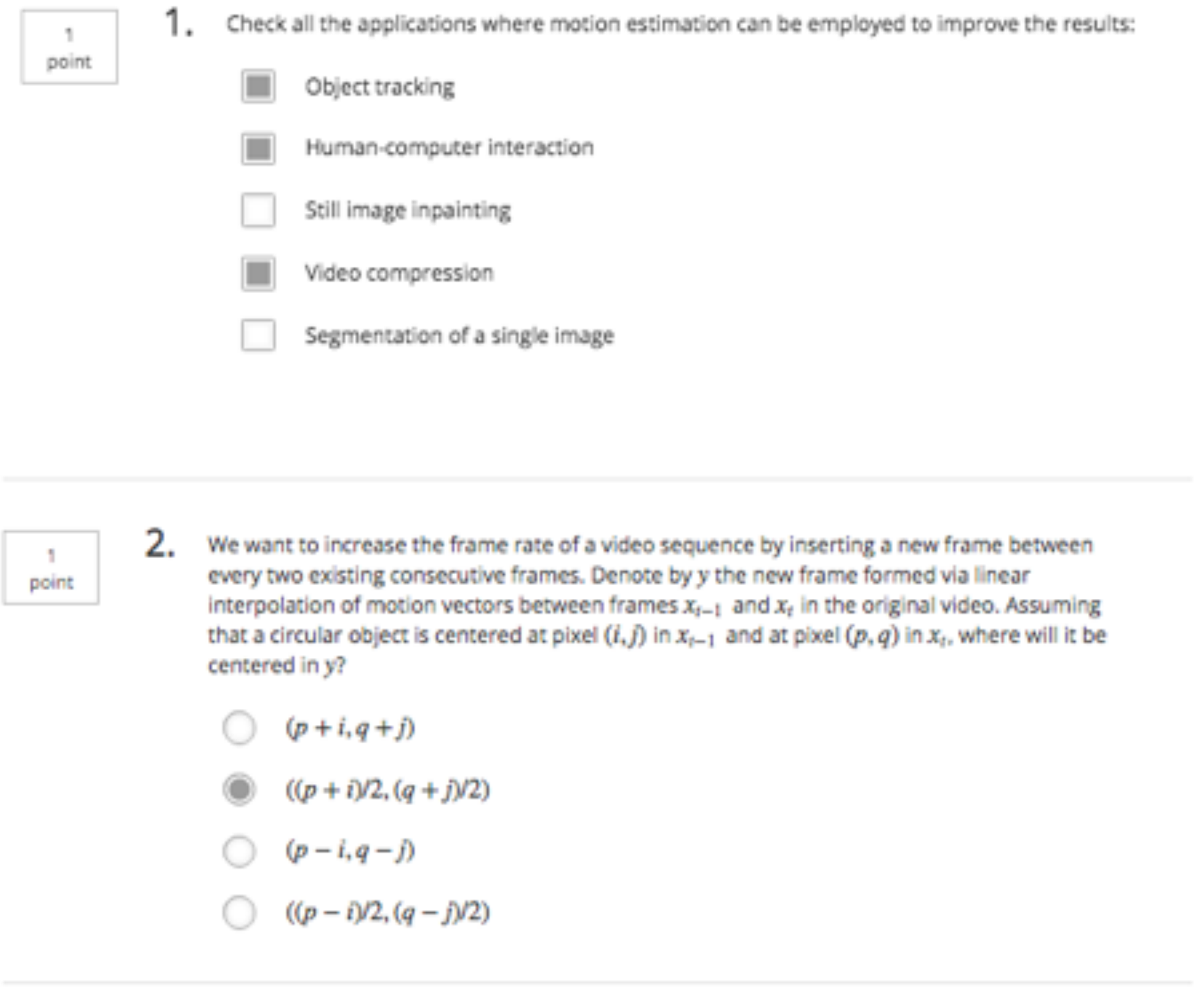
Convolutoin Examples:i9;;

Upsampling Images

Downsampling Images

Gaussian Pyramid:
Laplace Pyramid:

week 4 questions



1 point

1. Check all the applications where motion estimation can be employed to improve the results:

☑ Object tracking

☑ Human-computer interaction

□ Still image inpainting

☑ Video compression

□ Segmentation of a single image

1 point

2. We want to increase the frame rate of a video sequence by inserting a new frame between every two existing consecutive frames. Denote by y the new frame formed via linear interpolation of motion vectors between frames x_{t-1} and x_t in the original video. Assuming that a circular object is centered at pixel (i, j) in x_{t-1} and at pixel (p, q) in x_t, where will it be centered in y?

○ $(p + i, q + j)$

● $((p + i)/2, (q + j)/2)$

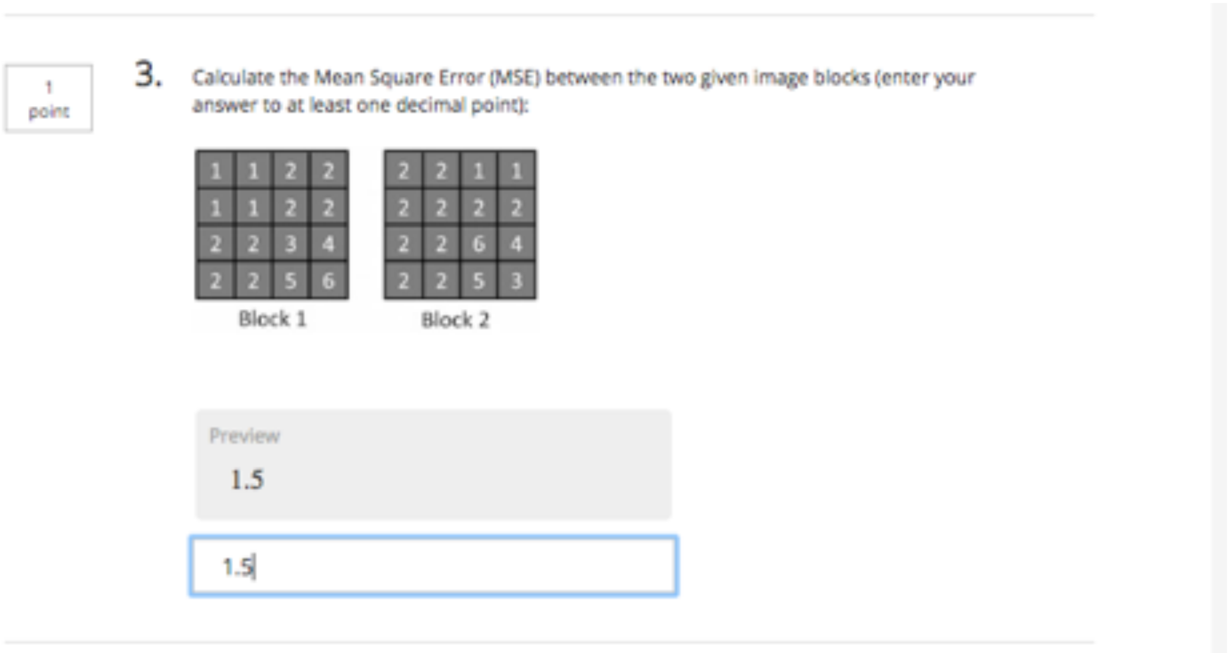
○ $(p - i, q - j)$

○ $((p - i)/2, (q - j)/2)$

Test w 2 points; (2,3), (4,5). To interpolate half way point (3,4); (x+y)/2 to get 3 and 4 from ponts above

Pick second dot

MSE:



1 point

3. Calculate the Mean Square Error (MSE) between the two given image blocks (enter your answer to at least one decimal point):

1	1	2	2
1	1	2	2
2	2	3	4
2	2	5	6

Block 1

2	2	1	1
2	2	2	2
2	2	6	4
2	2	5	3

Block 2

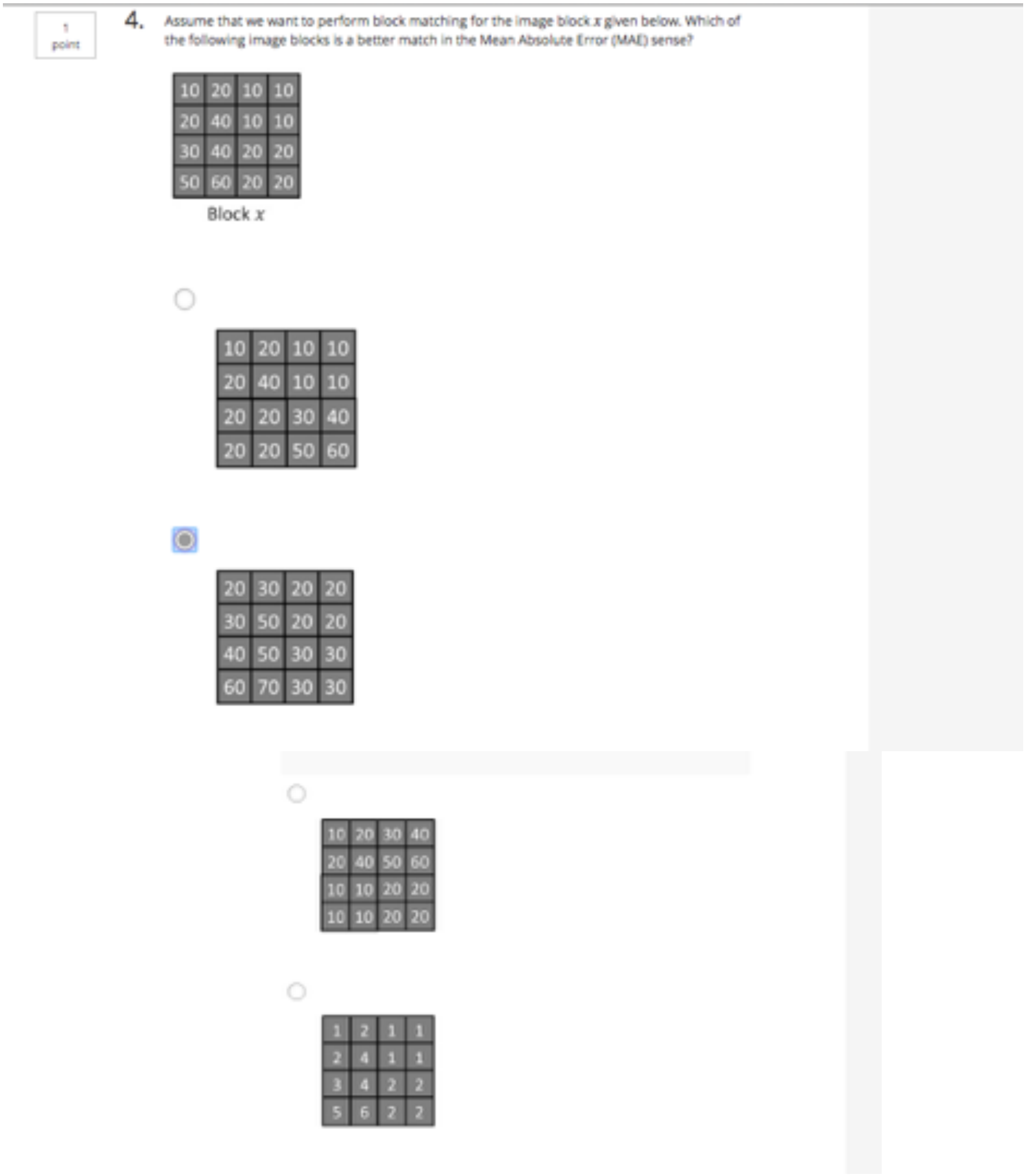
Preview

1.5

1.5

```
%calculate MSE between 2 matrices
X=[1 1 2 2; 1 1 2 2; 2 2 3 4; 2 2 5 6]
Y = [2 2 1 1; 2 2 2 2; 2 2 6 4; 2 2 5 3]
```

```
distxy=abs(X-Y).^2
msexy=sum(distxy(:)/numel(X))


%msexy =

%    1.5000
```



1 point

4. Assume that we want to perform block matching for the image block *x* given below. Which of the following image blocks is a better match in the Mean Absolute Error (MAE) sense?

10	20	10	10
20	40	10	10
30	40	20	20
50	60	20	20

Block *x*

○

10	20	10	10
20	40	10	10
20	20	30	40
20	20	50	60

●

20	30	20	20
30	50	20	20
40	50	30	30
60	70	30	30

○

10	20	30	40
20	40	50	60
10	10	20	20
10	10	20	20

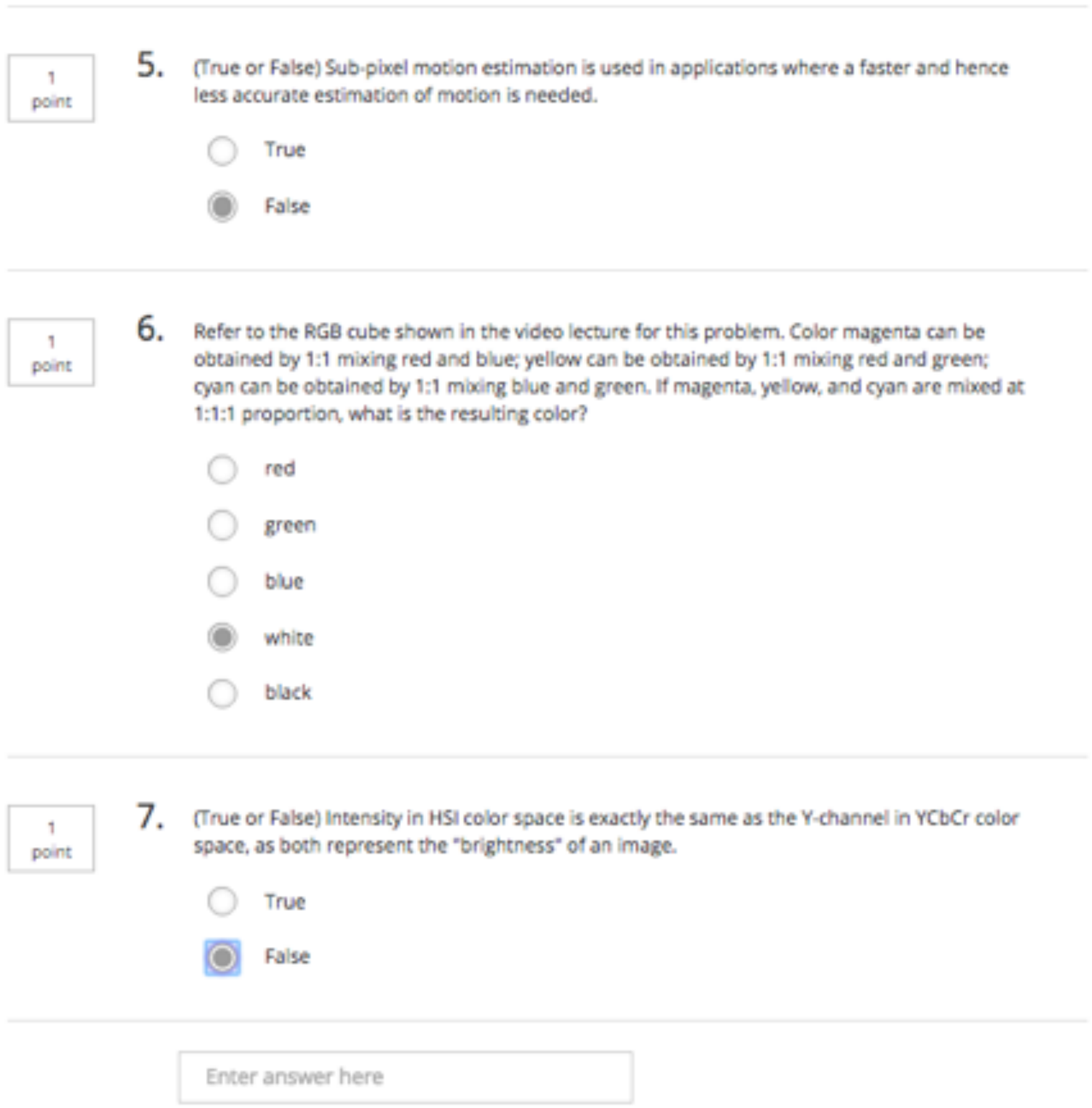
○

1	2	1	1
2	4	1	1
3	4	2	2
5	6	2	2

```
%MAE calculation example
mref=[10 20 10 10; 20 40 10 10; 30 40 20 20; 50 60 20 20]
m1=[10 20 10 10; 20 40 10 10; 20 20 30 40; 20 20 50 60]
m2=[20 30 20 20; 30 50 20 20; 40 50 30 30; 60 70 30 30]
m3=[10 20 30 40; 20 40 50 60; 10 10 20 20; 10 10 20 20]
m4=[1 2 1 1; 2 4 1 1; 3 4 2 2; 5 6 2 2]

dist1 = abs(mref-m1)
dist2=abs(mref-m2)
dist3 = abs(mref-m3)
dist4=abs(mref-m4)

mae1=sum(dist1(:))
mae2=sum(dist2(:))
mae3=sum(dist3(:))
mae4=sum(dist4(:))
```

```
mae1 =

   200


mae2 =
   160


mae3 =

   280


mae4 =

   351
```

Smallest is MAE2@160



1 point

5. (True or False) Sub-pixel motion estimation is used in applications where a faster and hence less accurate estimation of motion is needed.

○ True

● False

1 point

6. Refer to the RGB cube shown in the video lecture for this problem. Color magenta can be obtained by 1:1 mixing red and blue; yellow can be obtained by 1:1 mixing red and green; cyan can be obtained by 1:1 mixing blue and green. If magenta, yellow, and cyan are mixed at 1:1:1 proportion, what is the resulting color?

○ red

○ green

○ blue

● white

○ black

1 point

7. (True or False) Intensity in HSI color space is exactly the same as the Y-channel in YCbCr color space, as both represent the "brightness" of an image.

○ True

● False

Enter answer here

1 point

9. In the previous question, what was the corresponding MAE value (up to two decimal points)?

Enter answer here

Additional material Week 4: Optical Flow

From Forsythe/Ponce

http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_video/py_lucas_kanade/py_lucas_kanade.html

Feature Based Motion Estimation:
Scale Invariant Feature Transform: SIFT

Speeded up Robust Features: SURF

Kalman Filtering to predict motion of object

RGB -> YCrCB conversion:
rgb2gray supports the generation of C code using MATLAB® Coder™.
Algorithms
rgb2gray converts RGB values to grayscale values by forming a weighted sum of the R, G, and B components:

0.2989 * R + 0.5870 * G + 0.1140 * B
These are the same weights used by the rgb2ntsc function to compute the Y component.

convert RGB to HSV, is the grey scale in HSV same as Y?

RANSAC:
RANSAC used to
1) estimate the fundamental matrix of stereo vision
2) commonality between 2 sets of points for object detection
3) register sequential video frames for image stabilization(form of motion estimation)

What is a fundamental matrix?

Stereo Vision:
From
http://www.cc.gatech.edu/~afb/classes/CS4495-Spring2015-OMS/

Disparity Calculation: For multiple cameras, estimate …

Camera Calibration:

Week 5: Image Enhancement and Restoration
Restoring Images: Point Wise Intensity Transforms

Enhancement: modification of intesity to increase contrast
Restoration: deconvolution
Inpainting: enhancement or recovery
Smoothing/removal of noise: enhancement and restoration

Histogram Processing

Linear Noise Smoothing

Differentiate between image enhancement and image recovery. For image enhancement don't need to build a channel model or assume anything about the noise model. For image recovery; solving for H(s) starting w/AWGN and moving up.

Image enhancement involves spatial filtering and analyzing FFT.

1) Flat Filters

A flat filter

$$h(n1,n2)=\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

FFT(h(n1,n2))=

$$H(w1,w2) = \tfrac{1}{9}+\tfrac{2}{9}\cos(w1n1)+\tfrac{2}{9}\cos(w2n2)+\tfrac{2}{9}\cos(w1n1+w2n2)+\tfrac{2}{9}\cos(w1n1\text{-}w2n2)$$

Try different filter sizes, 3x3, 5x5. 7x7 and comparing results for quality by eye as to which one is better. Show the difference with filtered images and plots of FFT magnitude/phase. Narrowness of main lobe shows effedt of filter size on image.

This blurs images. One way to address this is to use spatially adaptive noise smoothing. Edge do nothing y)n1,n2)->x(n1,n2). because signman/signamnoise -> 0.

$$y(n1,n2) = \left(1 - \frac{\sigma_n^2}{\sigma_{xl}^2}\right) x(n1,n2) + \frac{\sigma_n^2}{\sigma_{xl}^2} xavg(n1,n)$$

where xavg is average image intensity, sigman is noise variance, sigma xl is image variancde.

compute the noise variance? not clear. the mean is from the image in a flat region like sky. Can figure out noise from an image patch with no edges. Use sky background.

edges:
flat regions:

The adaptive process is improved over standard flat filter.

this is combination of identity filter when see edges, see mean filter in flat regions.

Example of MATLAB script showing filtering in FFT/translating effects into spatial domain
NOT APPLICABLE TO CNN? CNN is time filtering vs. FFT.

I create an ADSL DMT symbol with only 3 tones.
tone1 and tone2 are part of the downstream band.
tone3 is not part of this band and should be filtered out.
DMT symbol is prepared in frequency domain.
Time signal is calculated by ifft.
Shown in figure 1.

By zero padding the time signal and using fft, the sidelobes of the rectangular
window become visible in figure 2.

Now let's create a high pass filter to filter out tone3.
Figure 3 shows characteristics of this filter.

Figure 4 shows filtered time signal.

First the filtering in red and then filtering again with correct initial conditions to skip the transitional phenomenon.

Now take fft of filtered time signal in top window of figure 5.
It is clear tone3 is filtered out, it's magnitude is 100 dB lower.
So far so good.

But now let's zero pad this filtered time signal.
And take the fft in bottom window of figure 5.
The side lobes in the low frequencies are still present.
tone3 is still filtered out, but not the sidelobes.
How do I get rid of the sidelobes?

Matlab script:

```
% dmt2.m
clear;
clc;

% dmt symbol with only 3 tones
fsymbol = 4312.5; % Hz
tone1 = 33;
tone2 = 34;
tone3 = 10; % to be filtered out
N = 512;
Ts = 1/(fsymbol*N);
tx = (0:N-1)*Ts;
i_fx = 0:N/2;
% create the tones
X = zeros(1,N);
X(tone1 + 1) = 1 * exp(1i*pi/4);
X(tone2 + 1) = 1 * exp(1i*(-pi*4));
X(tone3 + 1) = 1 * exp(1i*3*pi*4);
% make spectrum of real signal
X(N - tone1 + 1) = conj(X(tone1 + 1));
X(N - tone2 + 1) = conj(X(tone2 + 1));
X(N - tone3 + 1) = conj(X(tone3 + 1));
% get time domain
x = ifft(X);
max(imag(x))

% show dmt symbol in time and frequency domain
figure(1);
subplot(2,1,1);
plot(tx,x);
subplot(2,1,2);
XF = abs(X);
plot(i_fx,XF(i_fx+1),'xr');
```

```
% zero padded fft to show the side lobes
M = 16*N;
i_fy1 = 0:M/2;
i_fy2 = 0:M/N:M/2;
y = [x zeros(1,M-N)];
Y = 20*log10(abs(fft(y)));

figure(2);
plot(i_fy1,Y(i_fy1+1),'ob-');
hold on;
plot(i_fy2,Y(i_fy2+1),'+r');
hold off;
legend('zero padded signal','original tones');

% All frequency values are in Hz.
Fs = N*fsymbol;   % Sampling Frequency

Nf     = 150;     % Order
Fc     = 100000;  % Cutoff Frequency
DstopU = 1e-05;   % Upper Stopband Attenuation
DstopL = 1e-05;   % Lower Stopband Attenuation
DpassU = 0.001;   % Upper Passband Ripple
DpassL = 0.001;   % Lower Passband Ripple

% Calculate the coefficients using the FIRCLS function.
b  = fircls(Nf, [0 Fc Fs/2]/(Fs/2), [0 1], [DstopU 1+DpassU], [-DstopL ...
          1-DpassL]);
Hd = dfilt.dffir(b);
% show filter characteristics
freqz(Hd);

%z = filter(Hd,x);
[zd dd] = filter(b,1,x);
z = filter(b,1,x,dd);
figure(4);
plot(z,'xb-');
hold on;
plot(zd,'or-');
hold off;

Z = 20*log10(abs(fft(z)));
fz = 0:N/2;
z1 = [z zeros(1,M-N)];
Z1 = 20*log10(abs(fft(z1)));
fz1 = 0:M/2;
figure(5);
subplot(2,1,1);
plot(fz*fsymbol,Z(fz+1),'xr-');
```

```
subplot(2,1,2);
```

Reply
Posted by mnentwig ●November 9, 2014
Hi,

I can't run the script right now, lacking fircls() in octave. But from what you describe, I suspect the rectangular (aka Dirichlet) window is the problem. When the time domain signal is multiplied with the window, the spectra are convolved and this creates sidelobes.

Try a different window that better suits your purpose, i.e. hann() or blackmanharris().
The general tradeoff is center lobe width vs sidelobe decay rate. Details can be found here.
http://en.wikipedia.org/wiki/Window_function

Posted through www.DSPRelated.com
Reply
Posted by DougB ●November 17, 2014
>(I refer to the Matlab script below.)
>
>I create an ADSL DMT symbol with only 3 tones.
>tone1 and tone2 are part of the downstream band.
>tone3 is not part of this band and should be filtered out.
>DMT symbol is prepared in frequency domain.
>Time signal is calculated by ifft.
>Shown in figure 1.
>
>By zero padding the time signal and using fft, the sidelobes of the rectangular window become visible in figure 2.
>
>Now let's create a high pass filter to filter out tone3.
>Figure 3 shows characteristics of this filter.
>
>Figure 4 shows filtered time signal.
>First the filtering in red and then filtering again with correct initial conditions to skip the transitional phenomenon.
>
>Now take fft of filtered time signal in top window of figure 5.
>It is clear tone3 is filtered out, it's magnitude is 100 dB lower.
>So far so good.
>
>But now let's zero pad this filtered time signal.
>And take the fft in bottom window of figure 5.
>The side lobes in the low frequencies are still present.

>tone3 is still filtered out, but not the sidelobes.
>How do I get rid of the sidelobes?
>
>

You cannot filter out the sidelobes. In general the upstream and downstream are not orthogonal and thus they will interfere with each other even though they do not occupy the same set of tones. For upstream/downstream to be orthogonal, the ATU-R and ATU-C would have to adjust their symbol timing phase to compensate for the propagation delay across the wire - this is actually done in VDSL but not ADSL.

In order to demodulate you must remove the interfering sidelobes by estimating the interference and subtracting it out. This is done in an ADSL modem using hybrid analog/digital echo cancellation. There are many schemes for implementing this. Cancellation greater than 70 dB can be achieved.

-Doug

-Doug

2)

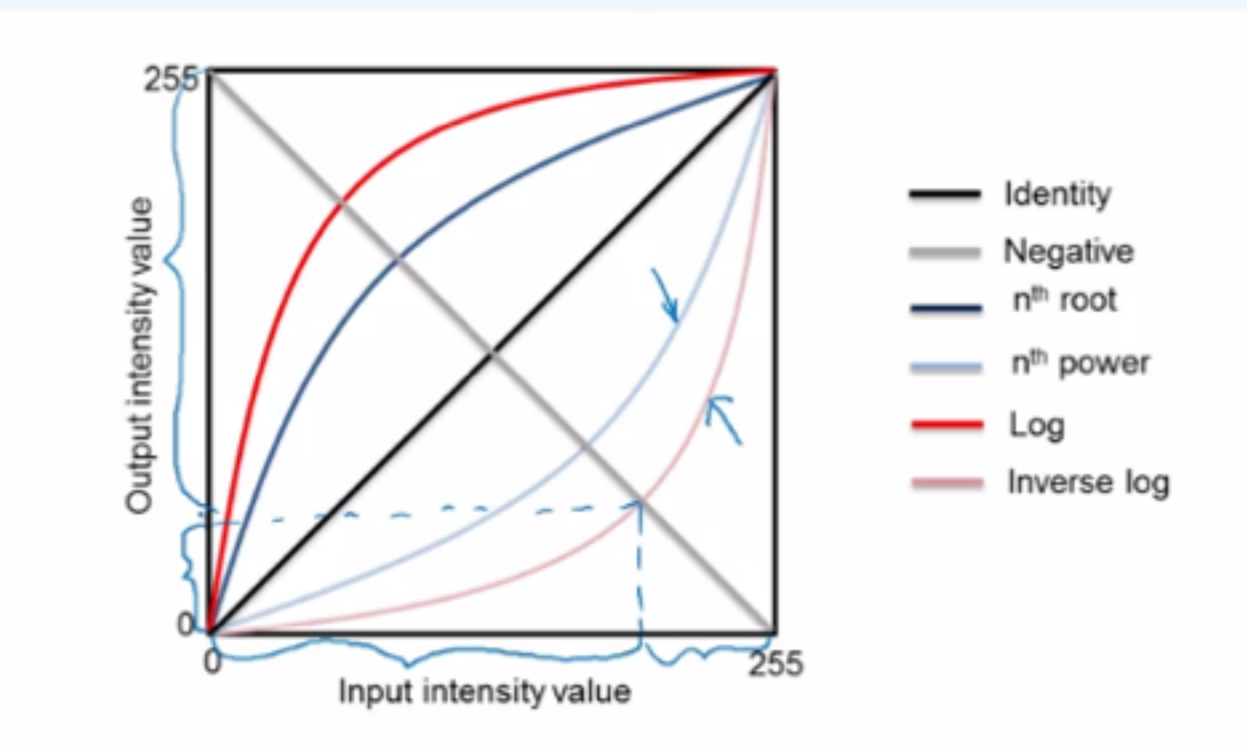
Non Linear Noise Smoothing
Sharpening
Homomorphic Filtering
Pseudo Coloring
Vide Enahancement

Point Wise Intensity TRANFORMATIONS

Single pixel:
y(n1,n2) = Tpoint(x(n2,n2))

Vector of pixels:
y(n1,n2) = **T**point(**x**(n2,n2))

Some graphs of intensity transforms, these can be used for histogram equalization:



Transforms which map input intensity to output intensity after transform.

The dark red and blue curve compresses the low and high input intensities; it compresses the range of small and large intensities into a smaller output range and it expands the middle intensities.

The light blue/red curves to the opposite; expanding the small input/output values and compressing the mid-range.

Histogram Equalization(source Computer Vision and Applications p. 95):

Goal is to make the existing intensity image distribution flat. If an image has intensities clustered at the lower end of the spectrum; the goal is to increase the intensities in the midrange and push the frequencies of the lower intensities down. We can do that with one of the intensity transforms.

To find an intensity mapping function f(I); this is the same problem as finding random samples from a probability density function.

Given an analogy of student test grades; how do we map a particular score to a grade to a given percentile; i.e. 50/100 becomse 75% where this is better than 75% of other test scores. Integrate the cumulative probability distribution:

$$c(i) = \int_{\square}^{\square} h(i)$$

the discrete form:

$$c(i)=\frac{1}{N}\sum_{i=0}^{I} h(i) = c(I\text{-}1)+\frac{1}{N}h(I)$$

The histogram of an image is the count of each intensity level. For a 256 gray level image, use a hashmap to calculate the count of intensities. N = number of pixels in image.

problem w/histogram equalizaiton is noise amplification of noise in dark regions

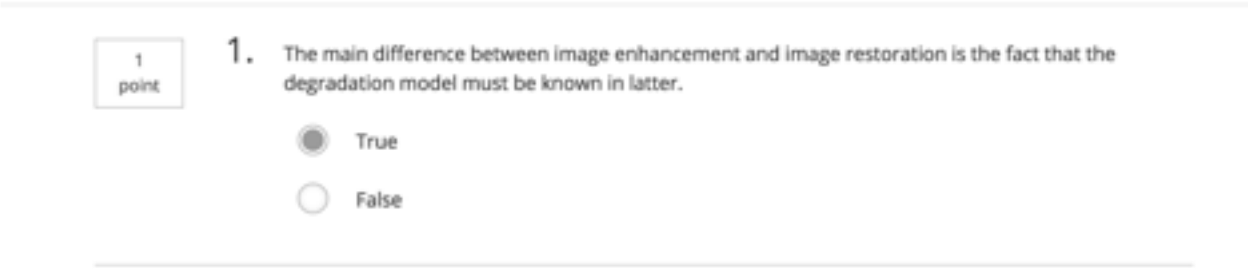
global histogram equalization(treating entire image as one single image) may not be better than some local approaches

1) histogram equalize blocks, divide image into blocks and equalize Problem is the block discontinuities;
2) moving window, N^2 in computation. Looks smoother. Don't see the image block discontinuities

Matlab:
OpenCV:

Linear Noise Smooting (denoising algoritm w/gaussian noise, compare vs. Weiner filter)
Non Linear Noise Smoothing
Sharpening (linear filter?)
Homomorphic Filtering
Pseudo Coloring
Vide Enahancement

Week 5:
Correct below:

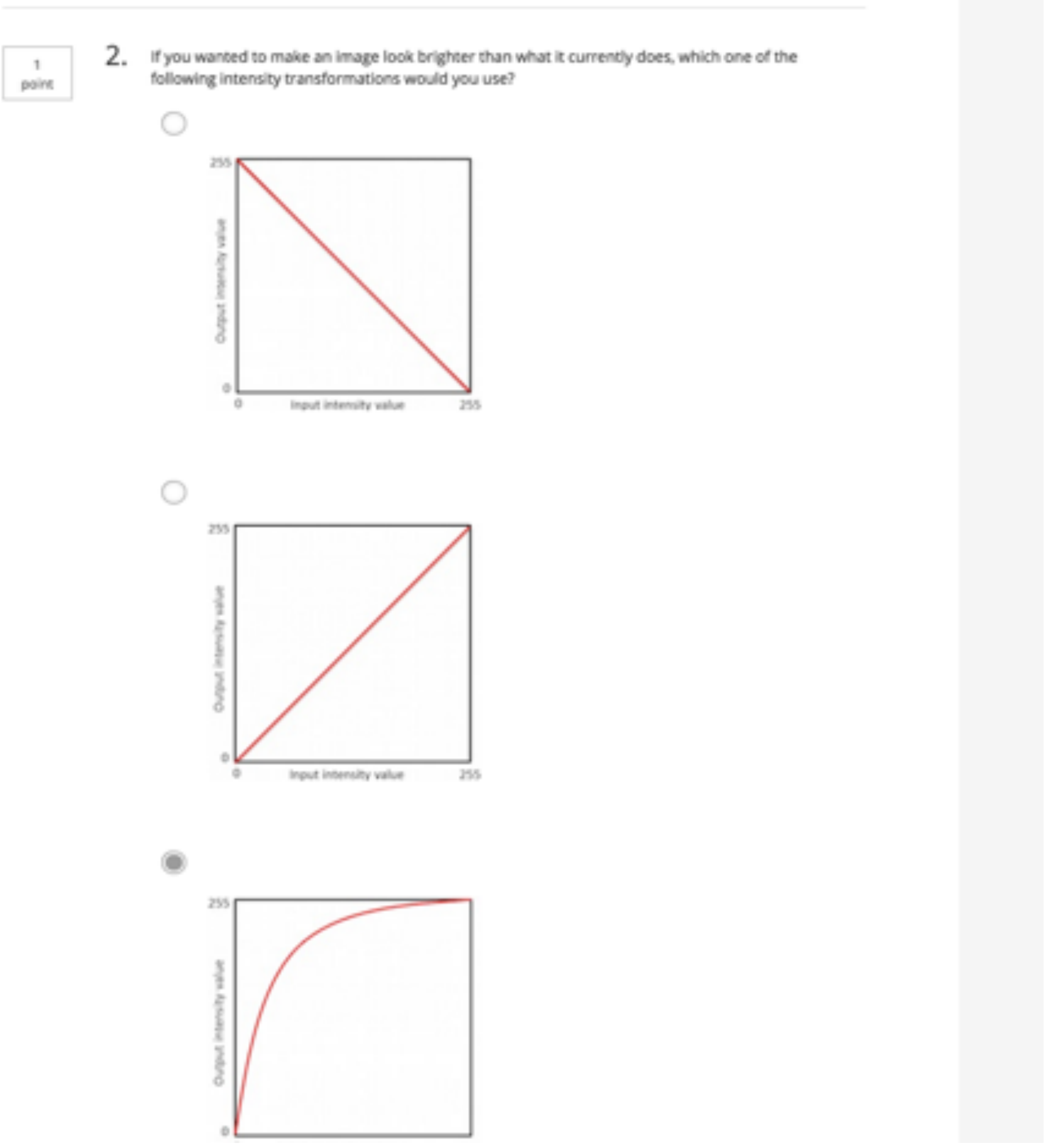


1 point

1. The main difference between image enhancement and image restoration is the fact that the degradation model must be known in latter.

● True

○ False



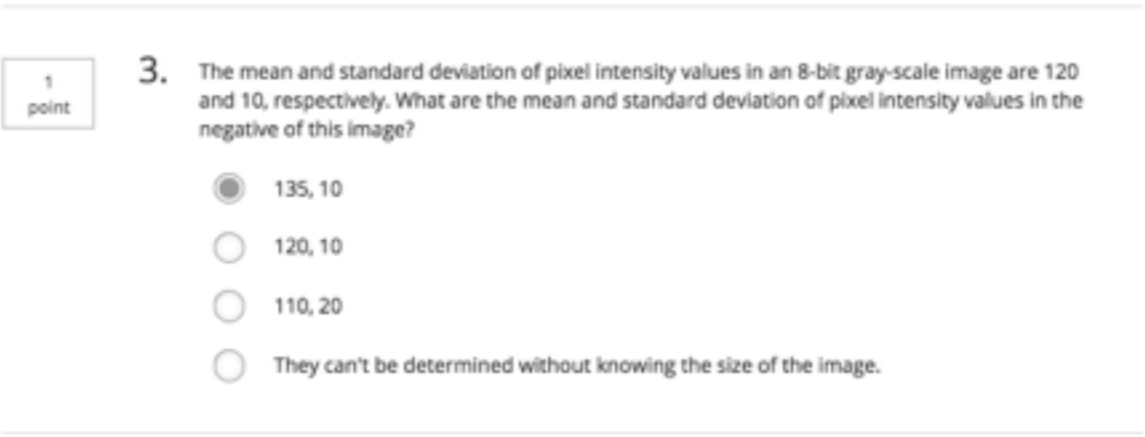
1 point

2. If you wanted to make an image look brighter than what it currently does, which one of the following intensity transformations would you use?

○

○

●



1 point

3. The mean and standard deviation of pixel intensity values in an 8-bit gray-scale image are 120 and 10, respectively. What are the mean and standard deviation of pixel intensity values in the negative of this image?

● 135, 10

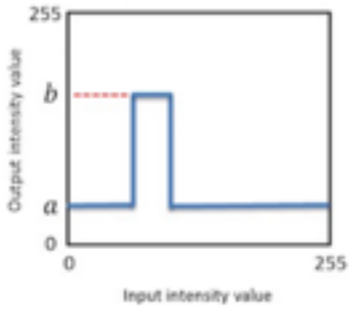
○ 120, 10

○ 110, 20

○ They can't be determined without knowing the size of the image.

1
point

4. Check all statements that apply to the following intensity transformation:



☒ The output image is binary since its pixels take on only two intensity levels.

□ It is possible to recover the input image after it undergoes this transformation.

☒ The mean intensity value of the pixels in the transformed image is always between a and b (including the end points a and b).

□ In the transformed image, the number of pixels with intensity level b is less than the number of pixels with intensity level a.

1
point

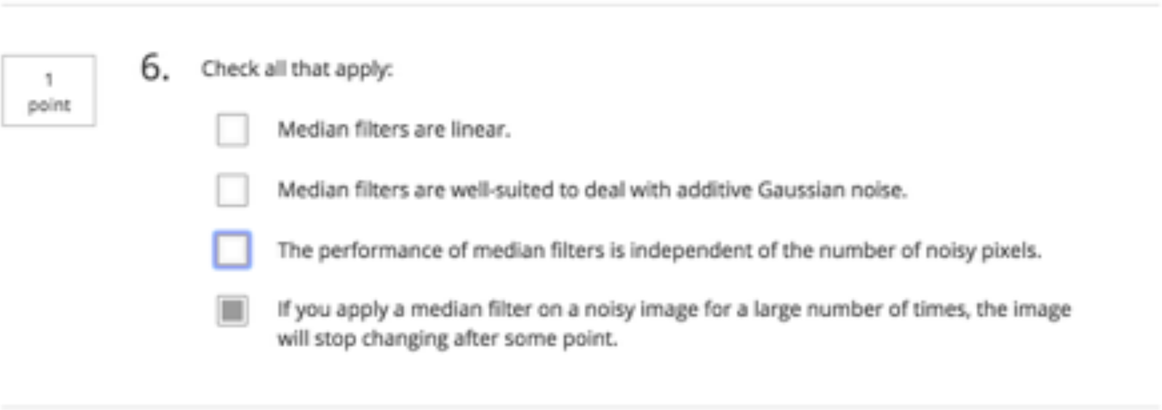
5. Check all statements that are true regarding image histogram equalization:

☒ After histogram equalization, the intensity values are more effectively distributed over the histogram range.

□ The mean intensity value of the pixels in an image increases after histogram equalization.

□ After histogram equalization images will always look brighter.

☒ If pixel p is the brightest pixel in image x, it will remain the brightest pixel after histogram equalization.



1 point

6. Check all that apply:

☐ Median filters are linear.

☐ Median filters are well-suited to deal with additive Gaussian noise.

☐ The performance of median filters is independent of the number of noisy pixels.

☒ If you apply a median filter on a noisy image for a large number of times, the image will stop changing after some point.

Week 6 Image Recovery:

There are both
Deterministic: output same with same input always.
Stochastic Restoration: input paramaters don’t produce same output every time; there is some random change

f -> H -> g

f: input; H transfer function, g output

if g is known ; blind recovery; reference blind deconvolution
if g and H partially known; partially blind

Blurring:

Blurring can be caused by a nonfocused camera, too slow a shutter speed, motion, bumps on road for a car camera, noise (thermal, etc..). One simple numerical approximation for blur is to take the 1 dimensional case and pick a simple example such as each pixel is affected by half the intensity from the pixel next to it…

$$x_1 \; x_2 \; x_3 \; x_4 \ldots$$

Some impulse response functions

For 1-D motions compensation:

$$h(i,j) = \frac{1}{L+1} \; for \; \frac{L}{2} \leq i \leq \frac{L}{2} \; 0 \; otherwise$$

Add motion at angle(search lit)

For Out of focus impulse response

$$h(i,j) = \frac{1}{\pi R} when \; \sqrt{i^2+j^2} < R \; 0 \; otherwise$$

Blurred SNR:
where M, N are sizes of the blurred and restored image.

$$BSNR = \frac{\frac{1}{MN} \sum_{i=0}^{M} \sum_{j=0}^{N} \left[g(i,j) - \overline{g}(i,j) \right]^2}{sigma^2}$$

$$g(i,j) = y(i,j) - n(i,j);\ y(i,j) = f(i,j) * * h(i,j)$$

$$\overline{g}(i,j) = E\{g(i,j)\}$$

$$sigma^2 = noise\ variance$$

Vector Notation for Images:

____
I
I
I
I
Toeplitz H
write image as vector X; take each row of image and turn into column vector.
transose(image(row))… transpose(image(row2) .. transpose(row N-1)

in vector notation 1dim convolution

y(n) = x(n) ** h(n) = SUM(X(k)h(n-k)

x(n) = [0,…N-1]
h(n) = [0 … L-1]
y(n) = [0 … N+L-2]

Circulant H:

Find Eigenvalues/Eigenvectors of circulant H.

Restored SNR:

To find the convolution with the above impulse respnoses you can either do this in spatial domain w/imfilter or using fft multiply.

http://www.mathworks.com/help/images/examples.html#btdrz6j-1 http://www.mathworks.com/help/images/examples/deblurring-images-using-a-regularized-filter.html?prodcode=ML

Inverse Filtering: This is the simplest example; figure out a h(i,j).

One example from: http://yuzhikov.com/articles/BlurredImagesRestoration1.htm
if we choose a blur which adds the pixel to the left to the x(i) pixel level:

x1+x0, x2+x1, x3+x2, x4+x3…. where x0 can be 0 or replicated

We can choose a restoration which subtracts pixels from the pixel on the left

x1+x0 (edge do nothing), x2+x1-x1-x0, x3+x2-x2+x0, x4+x3-x3-x0

becomes:

x1+x0 (edge do nothing), x2-x0, x3+x0, x4-x0

we get a pattern of alternating +/- x0 terms to the pixel values; assuming LTSI.

From the above deblurring process there are still artifacts.

There are 2 problems we are interested in; resoring the focus of the image to allow image classification; this may be different for CNNs than the human eye, and restoring the image enough to do motion estimation on for a sequence of blurred images.

In the spatial domain: use convolution to apply a deblurring transfer function. psf = fspecial('gaussian', 30, 8)
//defer graphs till later(use CN's code as template)
fspecial('motion', 40, 45);

Assume the blurring process is linear and includes an additional noise term:

g(x,y) = h(x,y) ** img(x,y) + n(x,y) //there are derivations we did in EE278 starting w/n(x,y) = Gaussian white noise where n(x,y) is a constant then making n(x,y) a gaussian distribution.

The discrete form of the above convolution is first w/o the noise term 2 summations:

Constrained Least Squares filter: as a solution for fixing small errors leading to blow up of noise term in inverse filtering.

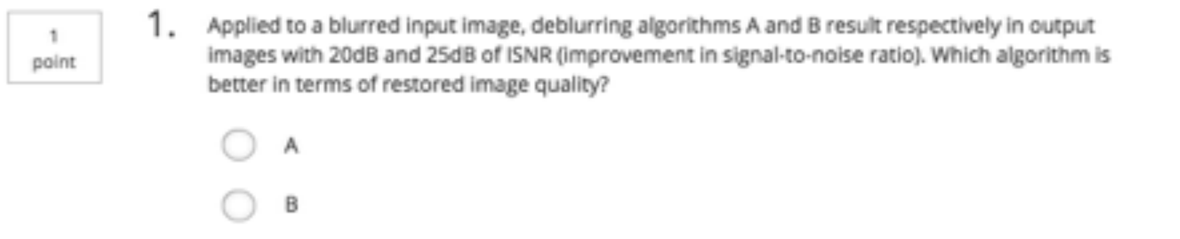
Regularization in CLS:

Iterative Approach Restoration:

Removing Ringing Artifacts:

Week 6 Quiz:

PSNR: is constant/MSE. The MSE is normalized between 0 - 1.
Smaller MSE => bigger PSNR. 25DB better. B better b/c smaller MSE?



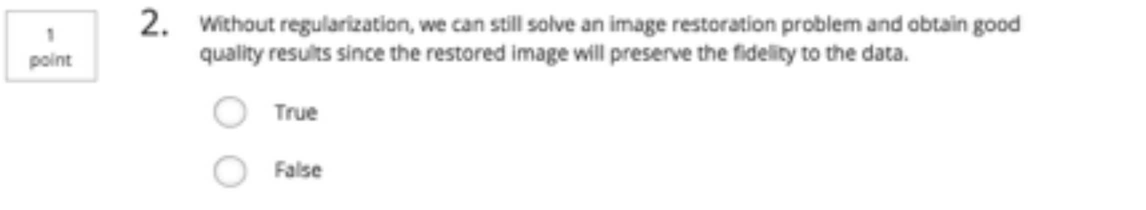
1 point

1. Applied to a blurred input image, deblurring algorithms A and B result respectively in output images with 20dB and 25dB of ISNR (improvement in signal-to-noise ratio). Which algorithm is better in terms of restored image quality?

○ A

○ B

1) 2
2) 2



1 point

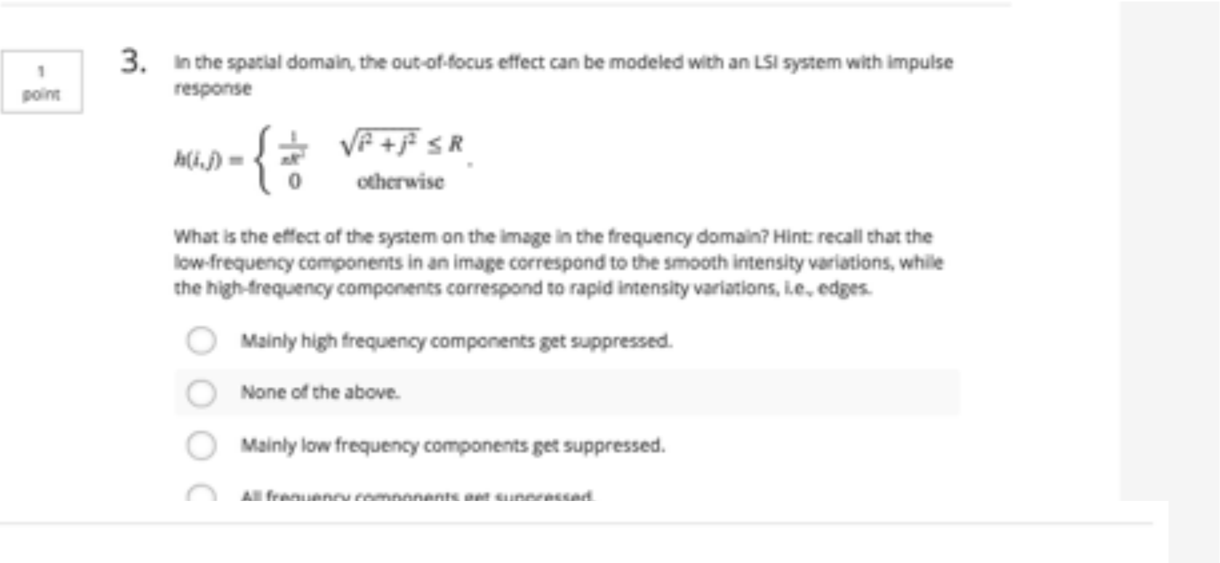
2. Without regularization, we can still solve an image restoration problem and obtain good quality results since the restored image will preserve the fidelity to the data.

○ True

○ False

Above is false. Regularization prevents overfitting.

3)out of focus effect removes edges or high freq componets 1
4) 3
5) t



1 point

3. In the spatial domain, the out-of-focus effect can be modeled with an LSI system with impulse response

$$h(i,j) = \begin{cases} \frac{1}{\pi R^2} & \sqrt{i^2 + j^2} \leq R \\ 0 & \text{otherwise} \end{cases}.$$

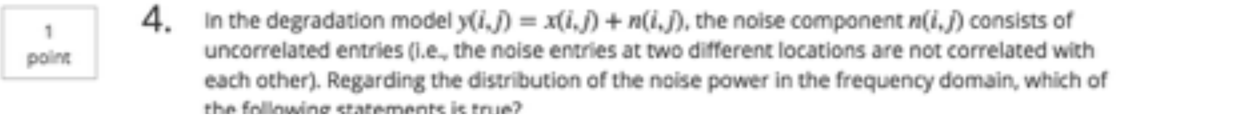
What is the effect of the system on the image in the frequency domain? Hint: recall that the low-frequency components in an image correspond to the smooth intensity variations, while the high-frequency components correspond to rapid intensity variations, i.e., edges.

○ Mainly high frequency components get suppressed.

○ None of the above.

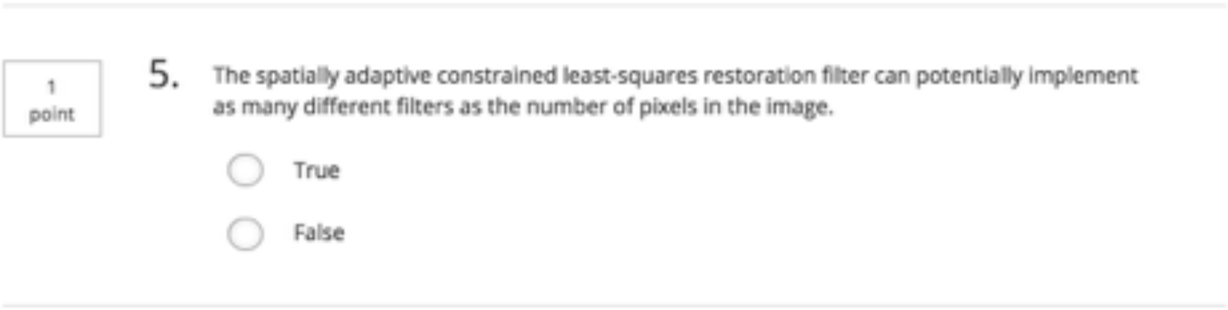
○ Mainly low frequency components get suppressed.

○ All frequency components get suppressed.



1 point

4. In the degradation model $y(i,j) = x(i,j) + n(i,j)$, the noise component $n(i,j)$ consists of uncorrelated entries (i.e., the noise entries at two different locations are not correlated with each other). Regarding the distribution of the noise power in the frequency domain, which of the following statements is true?

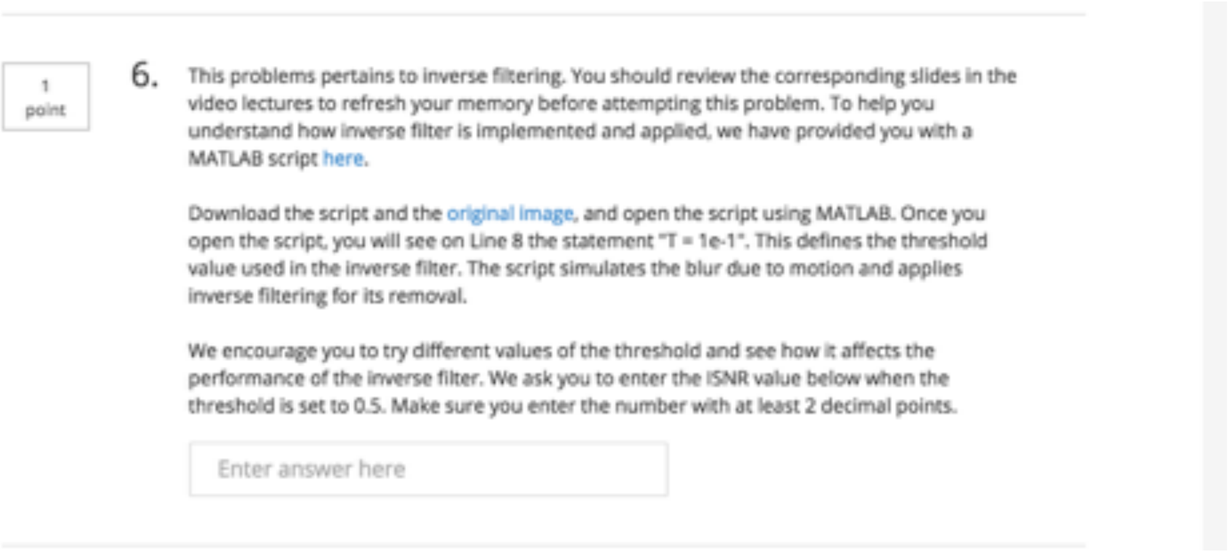


1 point

5. The spatially adaptive constrained least-squares restoration filter can potentially implement as many different filters as the number of pixels in the image.

○ True

○ False



1 point

6. This problems pertains to inverse filtering. You should review the corresponding slides in the video lectures to refresh your memory before attempting this problem. To help you understand how inverse filter is implemented and applied, we have provided you with a MATLAB script here.

Download the script and the original image, and open the script using MATLAB. Once you open the script, you will see on Line 8 the statement "T = 1e-1". This defines the threshold value used in the inverse filter. The script simulates the blur due to motion and applies inverse filtering for its removal.

We encourage you to try different values of the threshold and see how it affects the performance of the inverse filter. We ask you to enter the ISNR value below when the threshold is set to 0.5. Make sure you enter the number with at least 2 decimal points.

Enter answer here

Week 7:

Outline for Stochastic Image Restoration

Wiener Restoration Filter
Wiener Noise Smoothing Filter
Maximum Likelihood (ML)
Maximum A Posteriori Estimation (MAP)
Hierarchial Bayesian methods

Determinstic image restoration where we made assumptions like the image was smooth. Now we consider more powerful techniques where the image is considered a sample of a random field. A random field is a 2 dimensional random process.

Define Autocorrelation, Ergodicity, PSD of AC

Autocorrelation of a field f defines the correlation between values of a 2-d random process at 2 different times:

$$AC=\mathrm{E}[f(i,j)f\mathrm{conj}(k,l)]$$

f(k,l) is the complex conjugate of f(i,j). There are 4 index variables to indicate f() is taken at 2 different positions. If fconj is not complex and real then the complex conjugate of a real number is real. A specific AC calculation is one realization of the random field which represents this set of images. There are many realizations of this random field f which forms an ensemble of images. Assume you can perform multiple observations of this random field/images then you can calculate the expectation or mean/std deviation of the random field given certain assumptions.

Assume the random processes are WSS, wide sense stationary. The 1st moment and covariance don't vary with time. This also means we can shift the signal.

$$f(i,j,k,l) = f(i\text{-}k,\ j\text{-}l) = f(n1,n2)$$

where n1 and n2 are the distances between i,k and j,l. The WSS assumption allows us to know the autocorrelation and crosscorrelation from the average and std deviation of the data.

The random field to be modeled can be modeled as an isotropic exponential decay.

$$R(n_1,n_2) = c\gamma^{-(|n1|+|n2|)}$$

The autocorrelation of the isotropic exponential decay is a constant c and constant gamma to the negative of the absolute value of n1 and n2. Given data, we fit c and gamma to R(n1,n2). Furtherv away 2 pixels are, the smaller the AC.

Ergodocity/to simplfiy to ensemble averages = spatial averages and allows us to calculate the AC expectation via a spatial average:

$$R_{yy}(n_1,n_2) = l\mathrm{im}_{n\text{->}\infty}\frac{1}{(2N+1)^2}\sum_{k_1=-N}^{N}\sum_{k_2=-N}^{N} f(k_1,k_2)f\mathrm{conj}(k_1\text{-}n_1,k_2\text{-}n_2)$$

Power Spectrum of AC:

PSD(n1,n2) = FFT(R(n1,n2))

Weiner Filter Derivation:

y(i,j) = h(i,j) ** f(i,j) + w(i,j)

h(i,j) = impulse response of degradation
f(i,j) = image; try to find estimate of original image f(i,j)
w(i,j) = noise
y(i,j) = observed image

f(i,j) is a random field, know ac of f(i,j), a/c of w(i,j) and cross correlation between f(i,j) and w(i,j).

festimate(i,j) = argmin(f(i,j))E(lerror(i,j)l^2) = argmin(f(i,j))(E(lf(i,j)-festimate(i,j))l^2)

Bayesian Probabilities/Statistics;
A different interpretation not directly related to the counting of frequency as being the same as probability; loosely related from propositional logic like we saw in phil 160. Don't see how this is really true..

Bayesian probability is called an evidential probability. There exists some prior probability and based on some new evidence the prior is updated to a posterior probability. The Bayesian interpretation has a set of rules to do this.

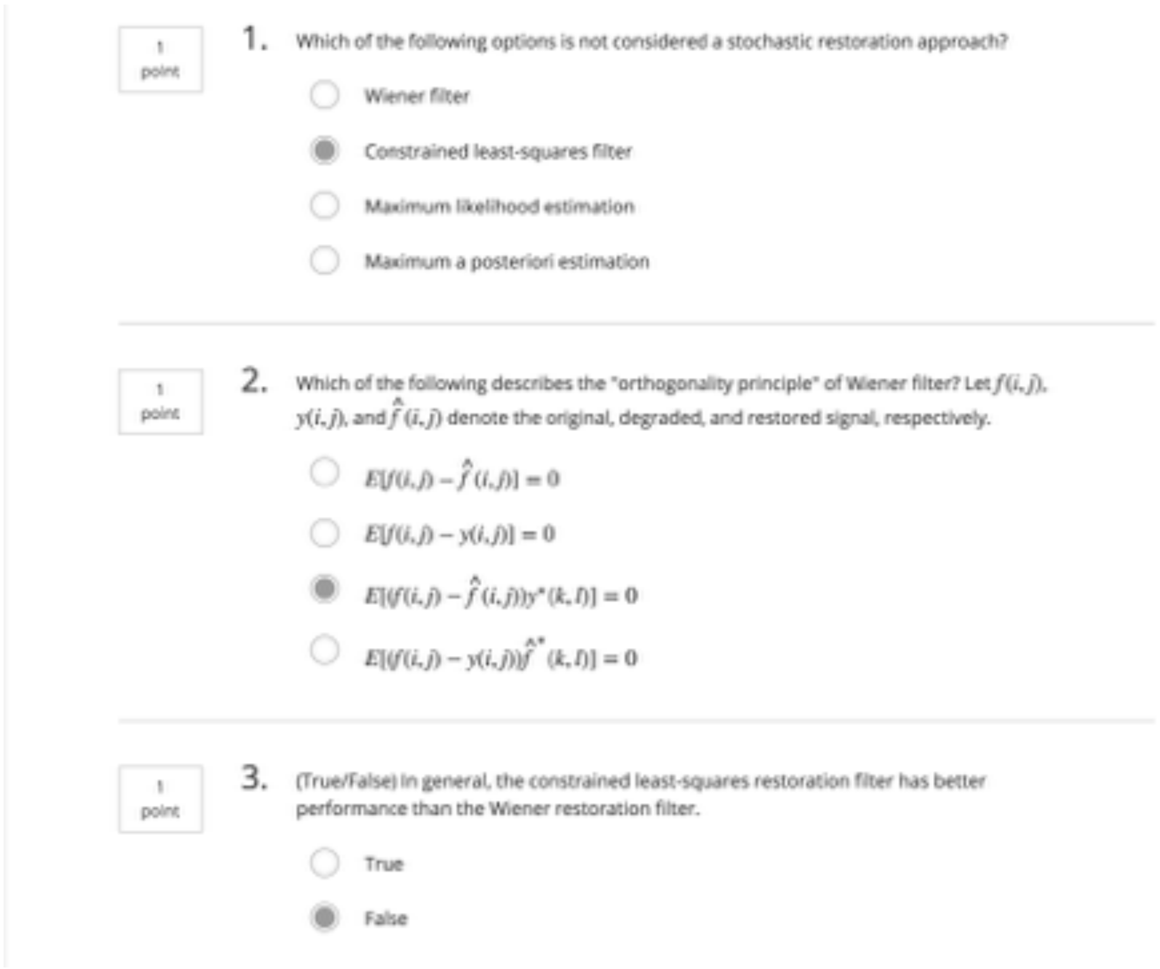
One method of constructing priors is th use Maximum Entropy. In ME, we use the probability distribution with the largest entropy.

Bayes Theorem describes how one observation may be related to another observation

p(f)

SuperResolution:

Week 7 Questions:



1 point

1. Which of the following options is not considered a stochastic restoration approach?

○ Wiener filter

● Constrained least-squares filter

○ Maximum likelihood estimation

○ Maximum a posteriori estimation

1 point

2. Which of the following describes the "orthogonality principle" of Wiener filter? Let $f(i,j)$, $y(i,j)$, and $\hat{f}(i,j)$ denote the original, degraded, and restored signal, respectively.

○ $E[f(i,j) - \hat{f}(i,j)] = 0$

○ $E[f(i,j) - y(i,j)] = 0$

● $E[(f(i,j) - \hat{f}(i,j))y^*(k,l)] = 0$

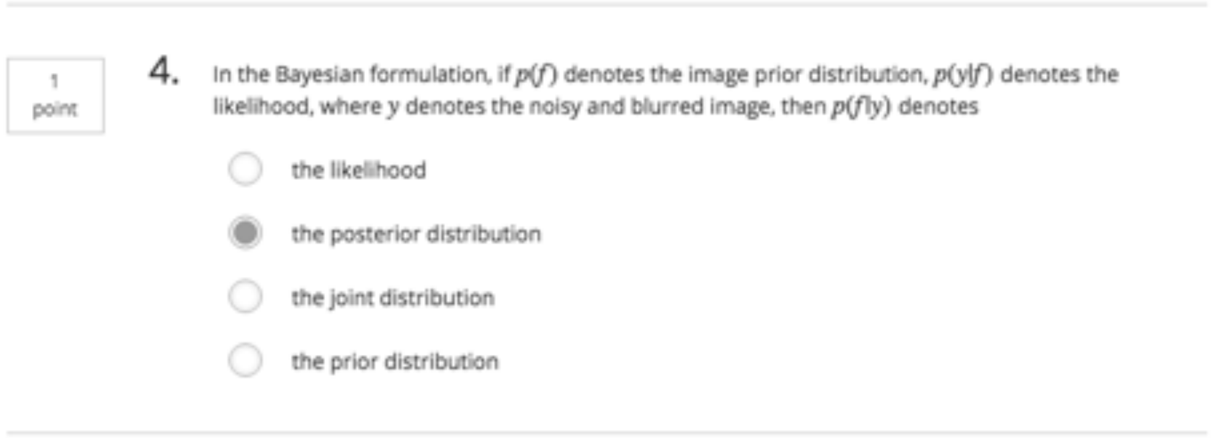
○ $E[(f(i,j) - y(i,j))\hat{f}^*(k,l)] = 0$

1 point

3. (True/False) In general, the constrained least-squares restoration filter has better performance than the Wiener restoration filter.

○ True

● False



1 point

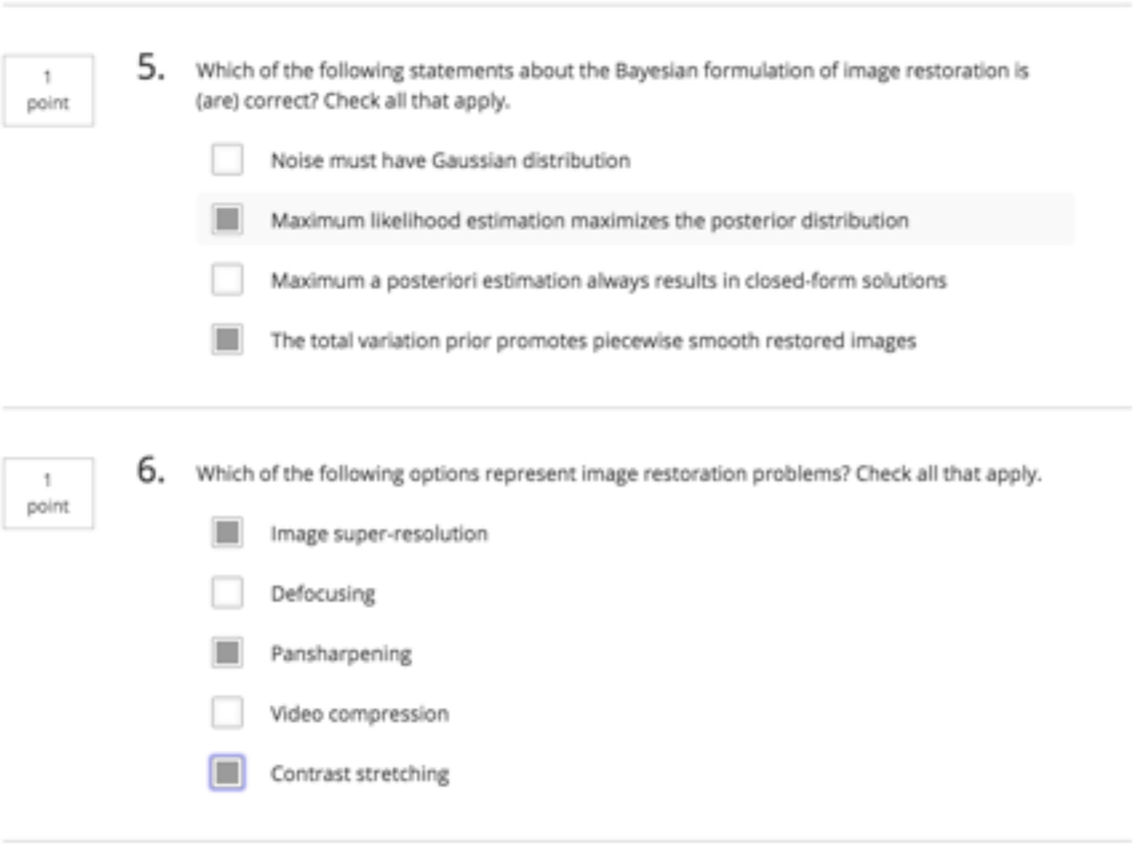
4. In the Bayesian formulation, if $p(f)$ denotes the image prior distribution, $p(y|f)$ denotes the likelihood, where y denotes the noisy and blurred image, then $p(f|y)$ denotes

○ the likelihood

● the posterior distribution

○ the joint distribution

○ the prior distribution



1 point

5. Which of the following statements about the Bayesian formulation of image restoration is (are) correct? Check all that apply.

□ Noise must have Gaussian distribution

☑ Maximum likelihood estimation maximizes the posterior distribution

□ Maximum a posteriori estimation always results in closed-form solutions

☑ The total variation prior promotes piecewise smooth restored images

1 point

6. Which of the following options represent image restoration problems? Check all that apply.

☑ Image super-resolution

□ Defocusing

☑ Pansharpening

□ Video compression

☑ Contrast stretching

#6 wrong, #7 not done

END:

Week 8:

OpenCV Mac Installation:

Build opencv. After the build you will have cv.so/cv2.so/opencv.hpp
Link these into XCode using Projects/Build Search Path for includes/libs
Test main.cpp program using Command Line project type.

Notes for later: demosaiking; restoring full color from incomplete color pixel information from CCD sensor

tracking blurred objects: mean shift tracking algorithm. His algotirm tracks the motion of the object.

Week 9:

Week 12:

Week 11:

Week 10: