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

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 $0 \le n_1, n_2 \le 2$ and zero elsewhere. Denote by $X(k_1, k_2)$, where $0 \le k_1, k_2 \le 2$, 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:

- [x] Object tracking
- [x] Human-computer interaction
- [ ] Still image inpainting
- [x] 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

Motion Estimation:

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 *[Math Processing Error]* 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

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

```
%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

1 point

8. In the next two problems you will perform block matching motion estimation between two consecutive video frames. Follow the instructions below to complete this problem.

(1) Download the two video frames from frame_1 and frame_2. The frames/images are of height 288 and width 352.

(2) Load the frame with file name "frame_1.jpg" into a 288×352 MATLAB array using function "imread", and then convert the array type from 8-bit integer to real number using function "double" or "cast" (note that the range of intensity values after conversion is between 0 and 255). Denote by I_1 the converted MATLAB array. Repeat this step for the frame with file name "frame_2.jpg" and denote the resulting MATLAB array by I_2. In this problem, I_2 corresponds to the current frame, and I_1 corresponds to the previous frame (i.e., the reference frame).

(3) Consider the 32×32 target block in I_2 that has its upper-left corner at $(65, 81)$ and lower-right corner at $(96, 112)$. Note this is MATLAB coordinate convention, i.e., the first number between the parenthesis is the row index extending from 1 to 288 and the second number is the column index extending from 1 to 352. This target block is therefore a 32×32 sub-array of I_2.

(4) Denote the target block by B_{target}. Motion estimation via block matching searches for the 32×32 sub-array of I_1 that is "most similar" to B_{target}. Recall in the video lectures we have introduced various forms of matching criteria, e.g., correlation coefficient, mean-squared-error (MSE), mean-absolute-error (MAE), etc.

In this problem, we use MAE as the matching criterion. Given two blocks B_1 and B_2 both of size $M \times N$, the MAE is defined as $MAE(B_1, B_2) = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} |B_1(i,j) - B_2(i,j)|$. To find the block in I_1 that is most similar to B_{target} in the MAE sense, you will need to scan through all the 32×32 blocks in I_1, compute the MAE between each of these blocks and B_{target}, and find the one that yields the smallest value of MAE.

Note in practice motion search is only performed over a certain region of the reference frame, but for the sake of simplicity, we perform motion search over the entire reference frame I_1 in this problem and the next. When you find the matched block in I_1, enter the sum of the x and y coordinates of the upper-left corner of the matched block in MATLAB convention. For example, if the matched block has the upper-left corner located at $(10, 20)$ then you must enter 30.

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: SIFT/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

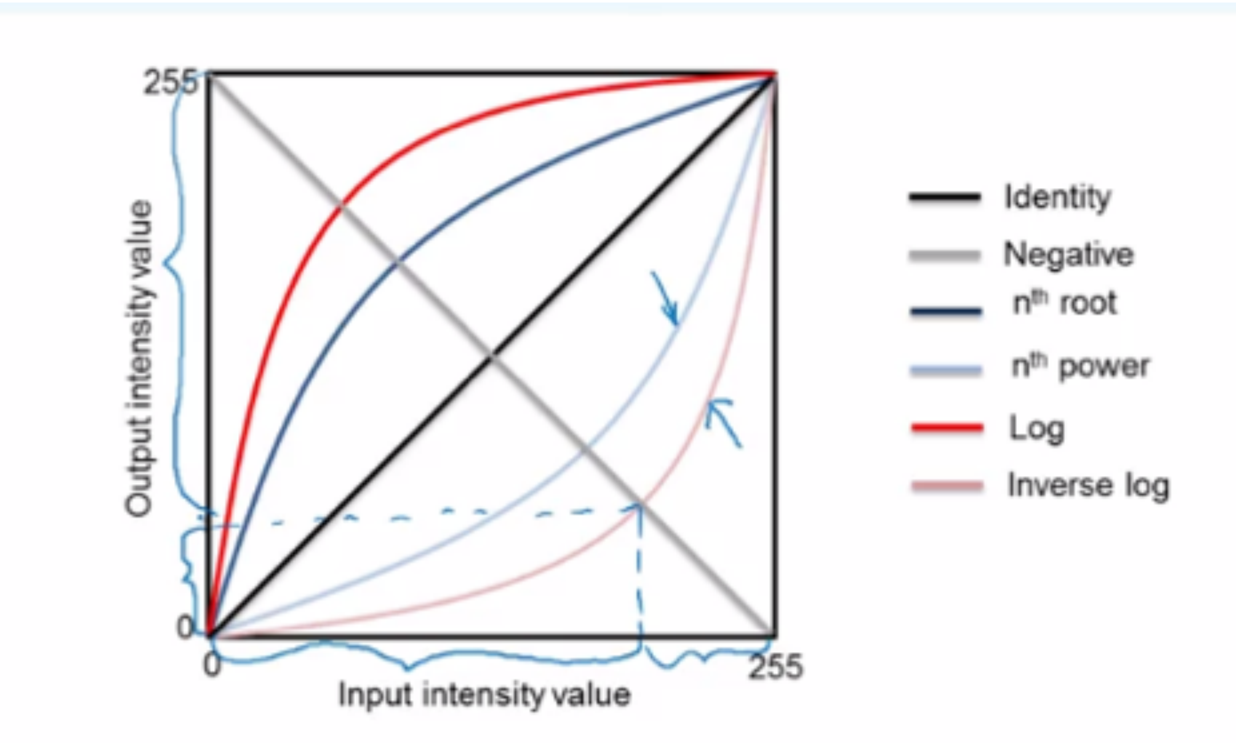
Matlab Image processing toolbox:

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:



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:

Week 5:
Correct below:

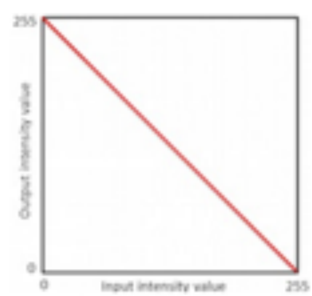
1 point

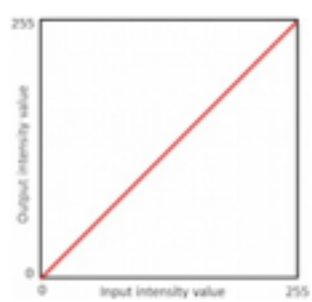
1. The main difference between image enhancement and image restoration is the fact that the degradation model must be known in latter.

- (selected) True
- False

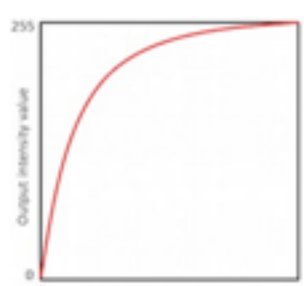
1 point

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?





(selected)



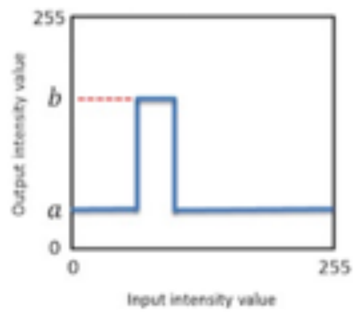
1
point

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:



- [x] 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.
- [x] 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:

- [x] 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.
- [x] 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.
- [x] 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.

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

$$g(i,j) = y(i,j) - n(i,j);\ y(i,j) = f(i,j) ** h(i,j)$$
$$\bar{g}(i,j) = E\{g(i,j)\}$$
$$sigma^2 = noise\ variance$$

Vector Notation for Images:

____
l
l
l
l
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:

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

END:

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.