The University *of* Texas

Health Science Center at Houston

School *of* Health Information Sciences

Noise and Filters

For students of HI 5323

“Image Processing”

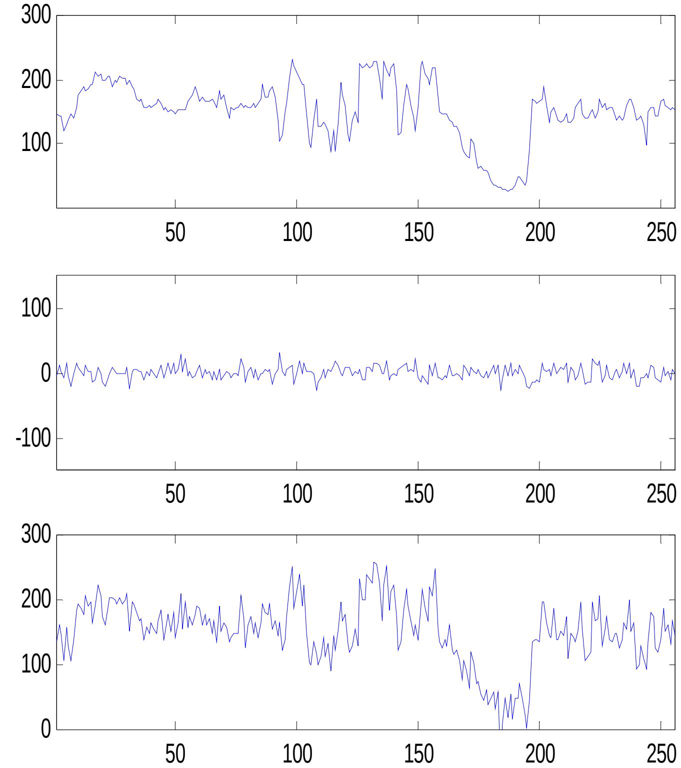
Willy Wriggers, Ph.D.

School of Health Information Sciences

http://biomachina.org/courses/processing/08.html

What is Noise?

- Anything that is NOT signal:
  - Signal is what carries information that we are interested in
  - Noise is anything else
- Noise may be
  - Completely random (both spatially and temporally)
  - Structured
  - Structured randomness



Statistical Review

Mean: The average or expected value

$$\mu = E\{x\} = \frac{1}{N}\sum x$$

Variance: The expected value of the squared error

$$\sigma^2 = E\{(x-\mu)\} = E\{x^2\} - \mu^2$$

Standard Deviation: The square root of the variance

$$\sigma = \sqrt{\sigma^2}$$

Covariance

The expected value of the product of the error between two elements of the signal:

$$\gamma_{ij} = E\{(x_i - \mu_i)(x_j - \mu_j)\}$$

This measures statistically the relationship between the error for the two elements:

$\gamma_{ij} = 0$	Independent (not related)
$\gamma_{ij} > 0$	Correlated (related)
$\gamma_{ij} < 0$	Inversely correlated (inversly related)

Ensembles of Images

Consider the picture $\tilde{I}(x)$ as a random variable from which we sample an ensemble of images from the space of all possibilities

This ensemble (or collection) of images has a mean (average) image, $\bar{I}(x)$

If we sample enough images, the ensemble mean approaches the noise-free original signal

- Often not feasible

Signal-To-Noise Ratio

If we compare the strength of a signal or image (the mean of the ensemble) to the variance between individual acquired images we get a signal-to-noise ratio:

$$SNR = \frac{\mu}{\sigma}$$

The better (higher) the SNR, the better our ability to discern the signal information

Problem: How to measure m to compute the SNR?

Covariance Matrix

We can build a matrix that contains the covariances between all samples:

$$C_{ij} = \gamma_{ij}$$

The diagonal elements are the individual sample variances:

$$C_{ii} = \sigma_i^2$$

A diagonal matrix indicates that the noise at each sample is independent of the others, i.e. uncorrelated

Non-zero diagonal elements of C indicate relationships between the noise at different positions, i.e. correlated

Additive Noise

Noise is often additive: causing the resulting signal to be sample-by-sample higher or lower than it should be

Such noise can be modeled by:

$$\tilde{I}(x) = \bar{I}(x) + n(x)$$

Poisson Noise

- Poisson distribution:

$$p(x;\lambda) = \frac{e^{-\lambda}\lambda^{x}}{x!}$$

  - Related to the quantum nature of light and matter
  - For discrete values
  - Applies only to non-negative values
  - Variance equal to the mean
  - Approaches a normal (Gaussian) distribution as the mean gets larger
  - Because the mean value changes for each pixel, the variance of each pixel is different

White Noise

Many noise process can be modeled by a normal (Gaussian)

Unlike Poisson noise, Gaussian-distributed noise is usually uniform over the image

Noise $\tilde{n}(x)$ that is

- Gaussian-distributed
- Zero-mean
- Uncorrelated
- Additive

is called white noise

Noise and the Frequency Domain

Noisy input:

$$\tilde{I}(x) = \bar{I}(x) + \tilde{n}(x)$$

Spectrum of noisy input:

$$\mathcal{F}(\tilde{I}(x)) = \mathcal{F}(\bar{I}(x)) + \mathcal{F}(\tilde{n}(x))$$

- White noise has equally random amounts of all frequencies
- “Colored” noise has unequal amount for different frequencies
- Since signals often have more low frequencies than high, the effect of white noise is usually greatest for high frequencies

Noise and Systems

Spectrum of noisy input:

$$\mathcal{F}(\tilde{I}(x)) = \mathcal{F}(\bar{I}(x)) + \mathcal{F}(\tilde{n}(x))$$

Spectrum of system's output:

$$H\mathcal{F}(\tilde{I}(x)) = H\mathcal{F}(\bar{I}(x)) + H\mathcal{F}(\tilde{n}(x))$$

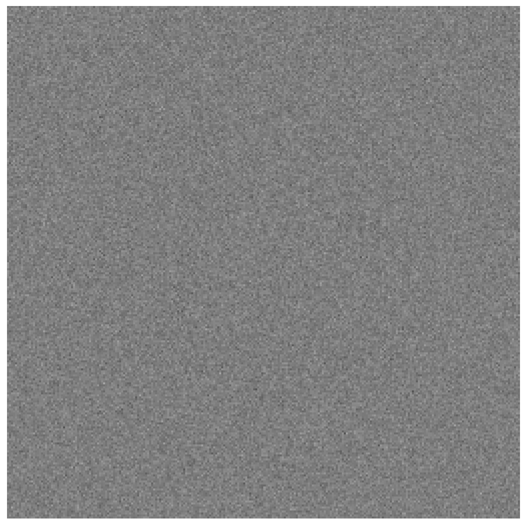
Solutions

- Filters
  - Low pass filter
    - eliminate high frequencies and leave the low frequencies.
  - High pass filter
    - eliminate low frequencies and leave high frequencies.
  - Band pass filter
    - only a limited range of frequencies remains
  - Gaussian smoothing
    - has the effect of cutting off the high frequency components of the frequency spectrum

Solutions

- Filters

+



=

image

noise

'grainy' image

Low-Pass Filter

- Recall that quick changes in a signal/image require high frequencies
- High frequency details are often “buried” in noise, which also requires high frequencies
- One method of reducing noise is pixel averaging:
  - Average same pixel over multiple images of same scene
  - Average multiple (neighboring) pixels in single image

Pixel Averaging

- Averaging multiple images not feasible if:
  - Object(s) in scene are moving
  - Only given a single image

- Averaging multiple (neighboring) pixels in a single image:
  - Gain: reduces noise
  - Cost: blurring

Noise Filtering

If an image is mainly low frequencies (with some high frequencies), white noise corrupts the high frequencies more than low

So, reduce the high frequency content of the noisy signal through low-pass filtering

Low-Pass Filtering = Spatial Blurring

Low-pass filtering and spatial blurring are the same thing

Any convolution kernel with all positive (or all negative) weights does:

- Weighted averaging
- Spatial blurring
- Low-pass filtering

They are all equivalent

Filtering and Convolution

Two ways to think of general filtering:

- Spatial: Convolution by some spatial-domain kernel

- Frequency: Multiplication by some frequency-domain filter

Can implement/analyze either way

Low-Pass Filtering

Tradeoff:

Reduces Noise

but

Blurs Image

The worse the noise, the more you need to blur to remove it

Original

After Low-pass filtering

“Ideal” Low-Pass Filtering

For cutoff frequency u_c:

$$H(u) = \Pi(u / u_c) = \begin{cases} 1 & \text{if } |u| \leq u_c \\ 0 & \text{otherwise} \end{cases}$$

What is the corresponding convolution kernel?

What problem does this cause?

What could you do differently?

Better (Smoother) Low-Pass Filtering

Gentler ways of cutting off high frequencies:

- Hanning

$$H(u) = \begin{cases} 0.5 + 0.5\cos(\pi\pi / 2u_c) & \text{if } |u| \le u_c \\ 0 & \text{otherwise} \end{cases}$$

- Gaussian

$$H(u) = e^{-u^2 / 2u_c^2}$$

- Butterworth

$$H(u) = \frac{1}{1 + \left(u^2 / u_c^2 \right)^n}$$

n controls the sharpness of the cutoff

Sharpening

- Blurring is low-pass filtering, so de-blurring is high-pass filtering:
  - Explicit high-pass filtering
  - Unsharp Masking
  - Deconvolution
  - Edge Detection
- Tradeoff:
  - Reduces Blur

    but
  - Increases Noise

High-Pass Filtering

- "Ideal":

$$H(u) = 1 - \Pi(u/u_c) = \begin{cases} 0 & \text{if } |u| \le u_c \\ 1 & \text{otherwise} \end{cases}$$

- Flipped Butterworth:

$$H(u) = 1 - \frac{1}{1 + (u^2 / u_c^2)^n}$$

High-Pass Filtering vs. Low-Pass Filtering

After Low-pass filtering

Original

After High-pass filtering

Unsharp Masking

Unsharp masking is a technique for high-boost filtering

Procedure:

- Blur the image.
- Subtract from the original.
- Multiply by some weighting factor.
- Add back to the original.

$$I' = I + \alpha(I - I * g)$$

where I' is the original image, g is the smoothing (blurring) kernel, and I is the final (sharpened) image

Unsharp Masking: Frequency Domain

Blur the image

Subtract from the original

Multiply by a weighting factor

Add back to the original

$$I + a(I - I * g)$$

Low-pass Filter

Original – low = high pass

Scale high (passed) frequencies

Original + scaled high = high boost

$$\mathcal{F}(I) + \alpha(\mathcal{F}(I) - \mathcal{F}(I) \cdot G)$$

Unsharp Masking: Implementation

$$I + \alpha(I - I * g)$$

$$\frac{1}{9}\left[\begin{bmatrix} 0 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \alpha\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right)\right]$$

$$= \frac{1}{9}\begin{bmatrix} -\alpha & -\alpha & -\alpha \\ -\alpha & 9+8\alpha & -\alpha \\ -\alpha & -\alpha & -\alpha \end{bmatrix}$$

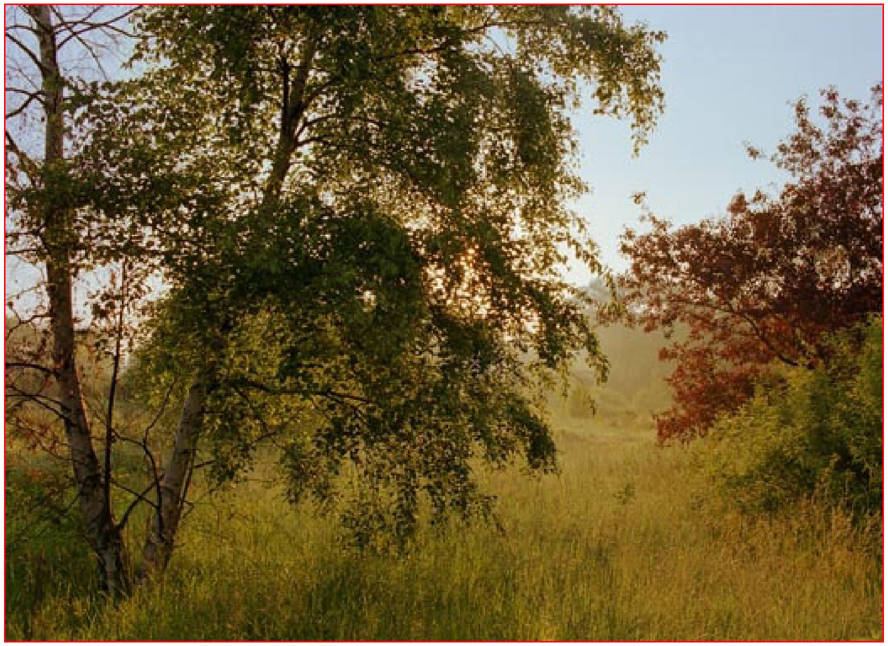
Unsharp Masking: Another Way

$$AI - I * g = (A - 1)I + (I - I * g)$$

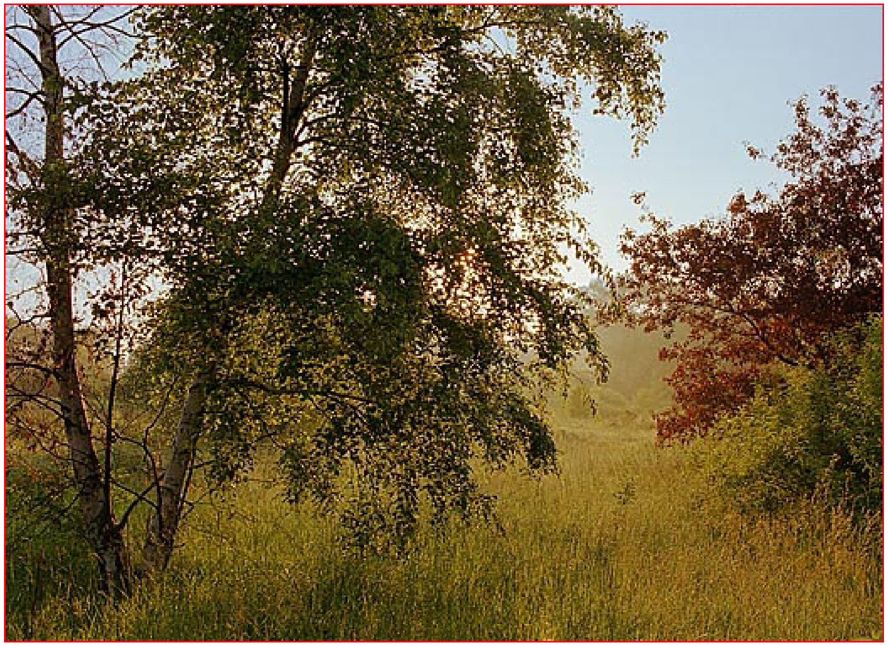
$$\frac{1}{9}\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 9A & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\right)$$

$$= \frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9A-1 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Unsharp Masking Image



Original Image



After Unsharp Masking

Deconvolution

If we want to "undo" low-pass filter $H(u)$,

$$H_{inv}(u) = 1 \big/ H(u)$$

Problem 1: This assumes you know the point-spread function

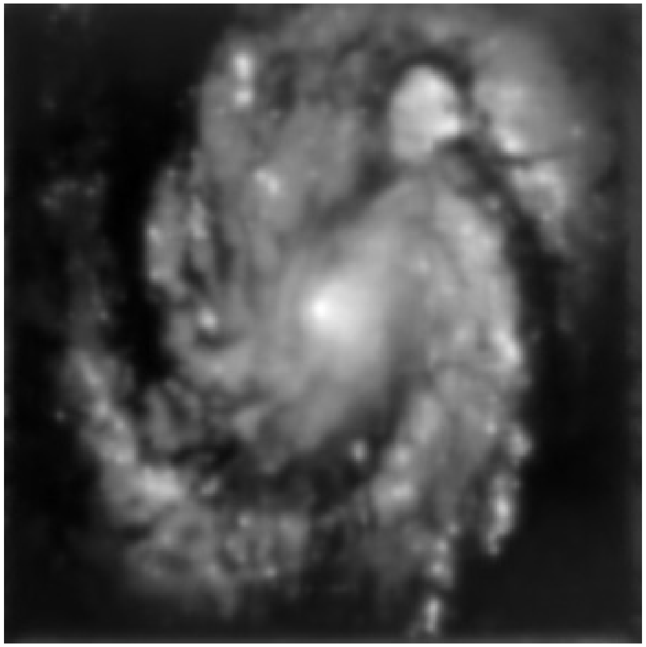
Problem 2: H may have had small values at high frequencies, so H_{inv} has large values (multipliers)

Small errors (noise, round-off, quantization, etc.) can get magnified greatly, especially at high frequencies

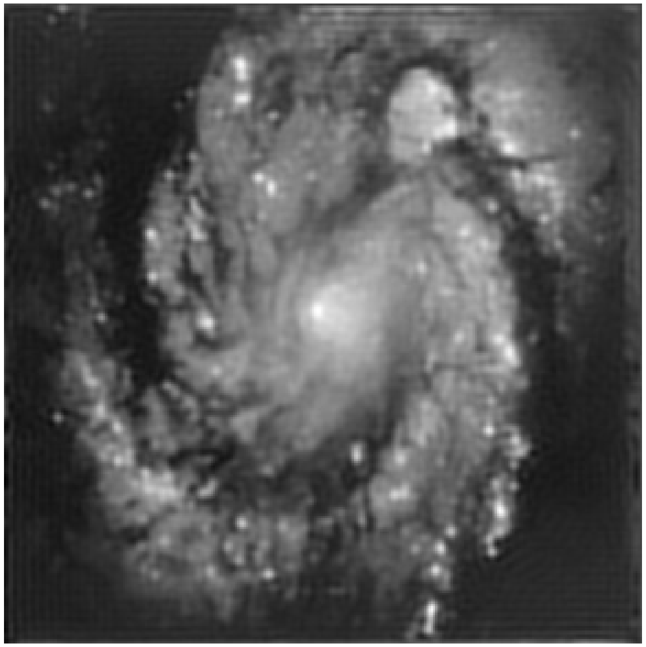
This is a common problem for all high-pass methods

Example: Deconvolution

Hubble space telescope image



Before deconvolution



After deconvolution

Band-Pass Filtering

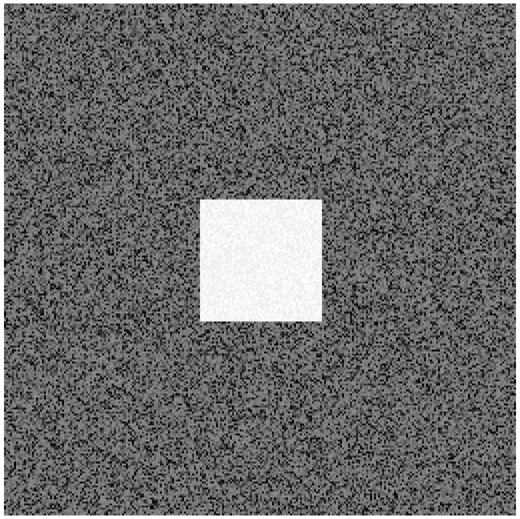
Tradeoff: Blurring vs. Noise

- Low-Pass: reduces noise but accentuates blurring
- High-Pass: reduces blurring but accentuates noise

A compromise:

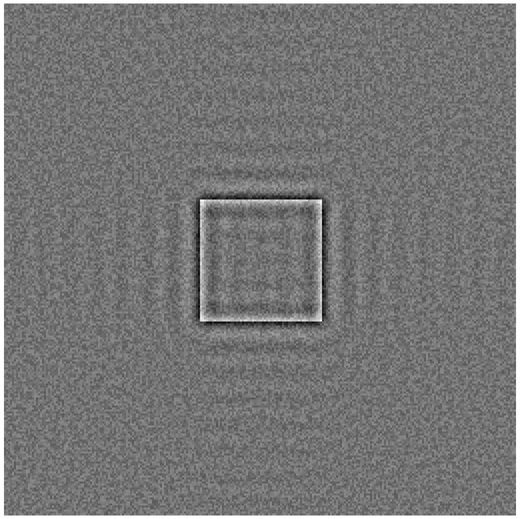
Band-pass filtering boosts certain midrange frequencies and partially corrects for blurring, but does not boost the very high (most noise corrupted) frequencies

Band-Pass Filtering vs. Low-Pass, High-Pass Filtering

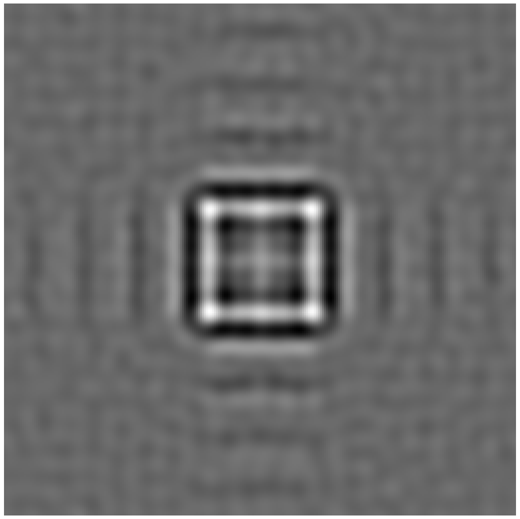


Original Image

After Low-pass filter



After High-pass filter



After Band-pass filter

Summary: Filtering

Consider what you want to do in the frequency domain

(the shape of the filter)

Frequency Domain	**Spatial/Temporal Domain**
Convert signal to/from frequency domain	Convert filter to equivalent convolution kernel/filter
Multiply in the frequency domain	Convolve in the spatial domain

Non-linear Operations

They are often mistakenly called “filters”

- Strictly speaking, non-linear operators are not filters

They can be useful, though

Examples:

- Order statistics (e.g., median filter)
- Iterative algorithms (e.g., CLEAN)
- Non-uniform convolution-like operations

Median "Filtering"

Instead of a local neighborhood weighted average, compute the *median* of the neighborhood

- Advantages:
  - Removes noise like low-pass filtering does
  - Value is from actual image values
  - Removes outliers – doesn't average (blur) them into result ("despeckling")
  - Edge preserving
- Disadvantages:
  - Not linear
  - Not shift invariant
  - Slower to compute

Median “Filtering”

Original image

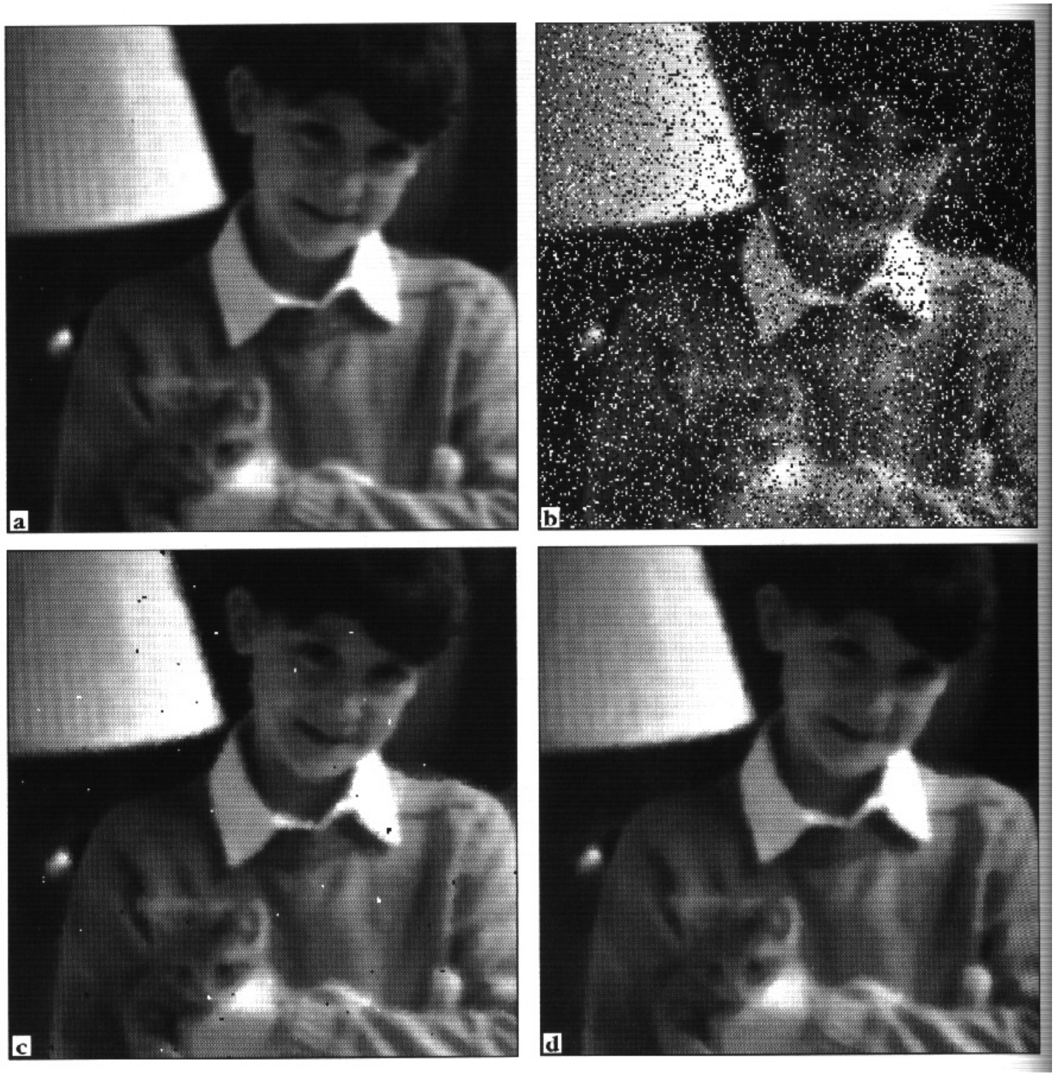


Image a with 10% of the pixels randomly selected and set to black, and another 10% randomly selected and set to white

Application of median filtering to image b using a 3x3 square region

Application of median filtering to image b using a 5x5 square region

Removal of shot noise with a median filter

Figure and Text Credits

Text and figures for this lecture were adapted in part from the following source, in agreement with the listed copyright statements:

http://web.engr.oregonstate.edu/~enm/cs519

Resources

Textbook:

Kenneth R. Castleman, Digital Image Processing, Chapter 11