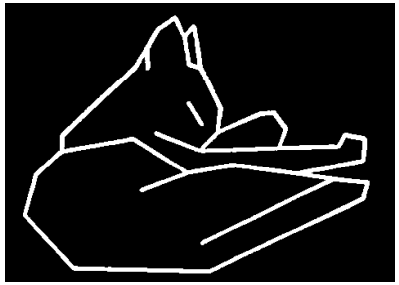
IMAGE PROCESSING

- Part of the following slides are from other courses on Computer Vision, available on the web, namely:
- CSc I6716 (Spring 2011), City College of New York, by Zhigang Zhu



Part I
Feature Extraction (1)

Image Enhancement

Zhigang Zhu, City College of New York zhu@cs.ccny.cuny.edu

- Local, meaningful, detectable parts of the image.

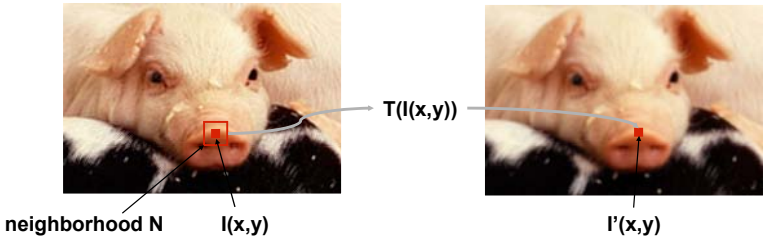
Topics

- Image Enhancement
  - Brightness mapping
  - Contrast stretching/enhancement
  - Histogram modification
  - Noise Reduction
  - ……..
- Mathematical Techniques
  - Convolution
  - Gaussian Filtering
- Edge and Line Detection and Extraction
- Region Segmentation
- Contour Extraction
- Corner Detection

Image Enhancement

- Goal: improve the 'visual quality' of the image
  - for human viewing
  - for subsequent processing
- Two typical methods
  - spatial domain techniques....
    - operate directly on image pixels
  - frequency domain techniques....
    - operate on the Fourier transform of the image
- No general theory of 'visual quality'
  - General assumption: if it looks better, it is better
  - Often not a good assumption

Spatial Domain Methods



- Transformation T
  - point - pixel to pixel
  - local - local area to pixel
  - global - output value at a specific coordinate depends on all values in the input image. (ex: DFT)
- Neighborhoods
  - typically rectangular
  - typically an odd size: 3x3, 5x5, etc
  - centered on pixel I(x,y)
- Many IP algorithms rely on this basic notion

$$I'(x,y) = T(I(x,y))$$

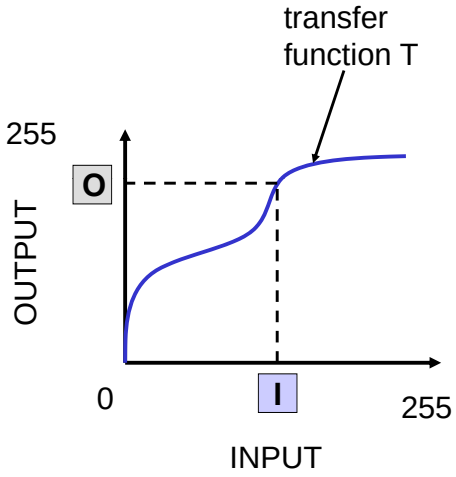
Point transformations

Histogram-based transformations

Point Transforms: general idea

$$O = T(I)$$

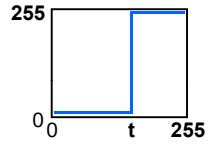
Input pixel value, I, mapped to output pixel value, O, via transfer function T.



Point Transforms:Thresholding

- T is a point-to-point transformation
  - only information at I(x,y) used to generate I'(x,y)
- Thresholding

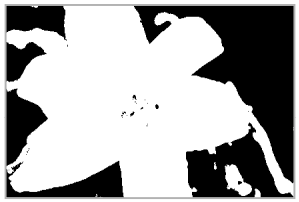
$$I'(x,y) = \begin{cases} I_{max} \text{ if } I(x,y) > t \\ I_{min} \text{ if } I(x,y) \le t \end{cases}$$

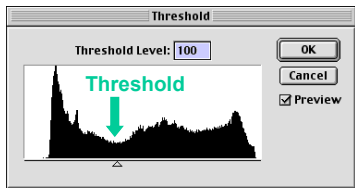


t=89

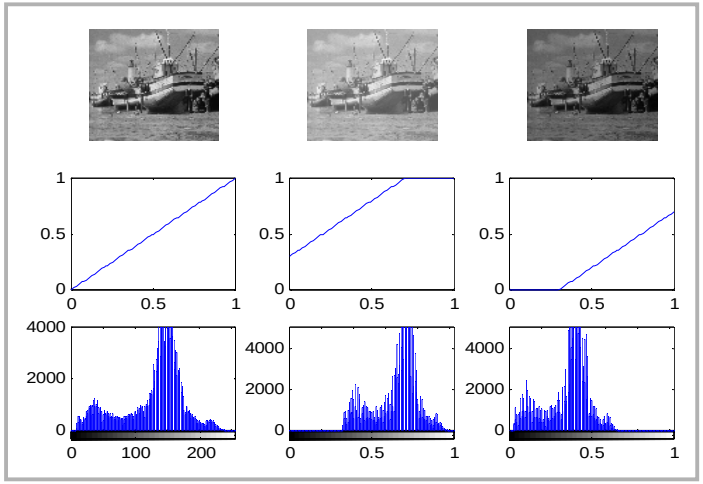
Threshold selection

- Arbitrary selection
  - select visually
- Use image histogram

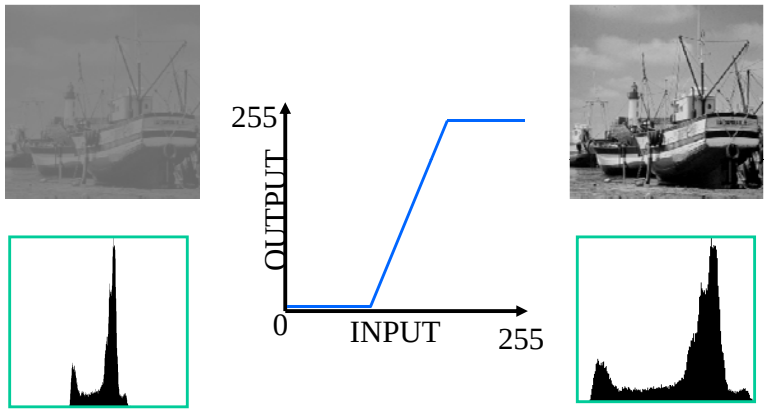




Point Transforms: Brightness

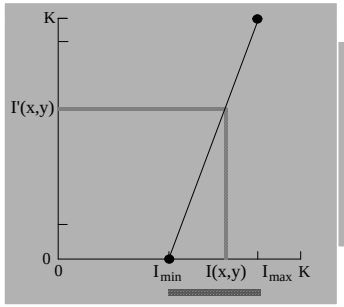


Point Transforms: Linear Stretch



Linear Scaling

- Consider the case where the original image only utilizes a small subset of the full range of gray values:
- New image uses full range of gray values.
- What's F? {just the equation of the straight line}



Input image I(x,y)
Gray scale range: [I_{min} , I_{max}]

Output image I'(x,y) = F [I(x,y)]
Desired gray scale range: [0 , K]

Linear Scaling

- F is the equation of the straight line going through the point (I_{min} , 0) and (I_{max} , K)

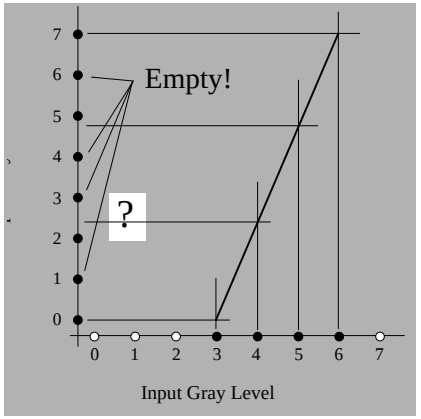
$$I'(x,y) = \frac{K}{I_{max} - I_{min}} I(x,y) - \frac{K}{I_{max} - I_{min}} I_{min}$$

$$I' = mI + b$$

- useful when the image gray values do not fill the available range.
- Implement via **lookup tables**

Scaling Discrete Images

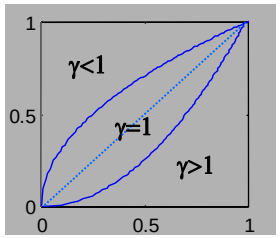
- Have assumed a continuous grayscale.
- What happens in the case of a discrete grayscale with K levels?



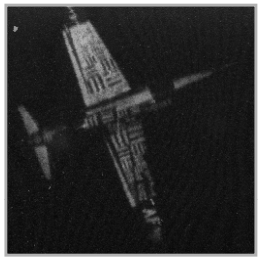
Non-Linear Scaling: power law

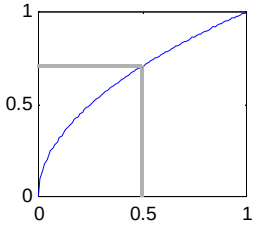
$$O = I^{\gamma}$$

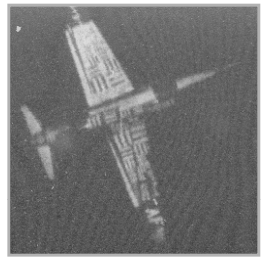
- $\gamma < 1$ to enhance contrast in dark regions
- $\gamma > 1$ to enhance contrast in bright regions.

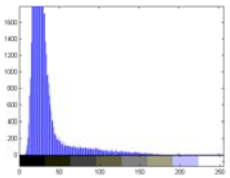


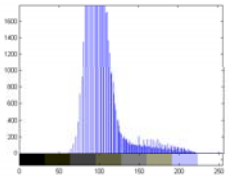
γ=0.5



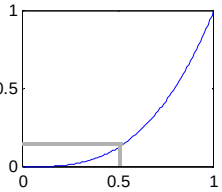


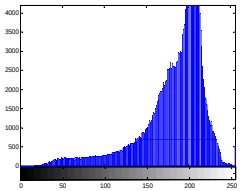


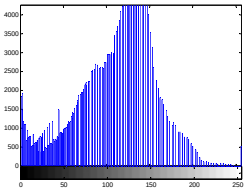




γ=3.0







Examples

- Technique can be applied to color images
  - same curve to all color bands
  - different curves to separate color bands:

  - but ... be careful, colors may become distorted...!

- Point transformations are usually applied using **LUT's (Look Up Tables)**

Histograms

- The image shows the spatial distribution of gray values.
- The image histogram discards the spatial information and shows the relative frequency of occurrence of the gray values.

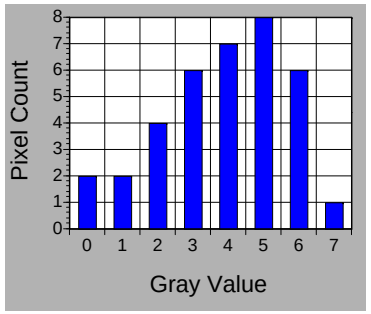
Image

0	3	3	2	5	5
1	1	0	3	4	5
2	2	2	4	4	4
3	3	4	4	5	5
3	4	5	5	6	6
7	6	6	6	6	5

Gray Value	Count	Rel. Freq.
0	2	.05
1	2	.05
2	4	.11
3	6	.17
4	7	.20
5	8	.22
6	6	.17
7	1	.03
Sum=	36	1.00

Image Histogram

- The histogram typically plots the absolute pixel count as a function of gray value:

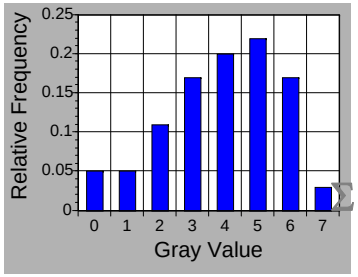


For an image with dimensions M by N

$$\sum_{i=I_{\min}}^{I_{\max}} H(i) = MN$$

Probability Interpretation

- The graph of relative frequency of occurrence as a function of gray value is also called a histogram:



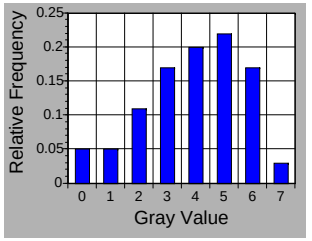
- Interpreting the relative frequency histogram as a probability distribution, then:

P(I(x,y) = i) = H(i)/(MxN)

Cumulative Density Function

- Interpreting the relative frequency histogram as a probability distribution, then:

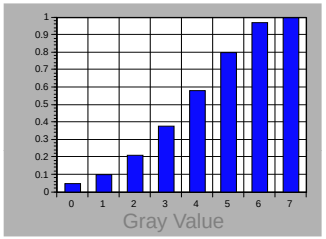
P(I(x,y) = i) = H(i)/(MxN)



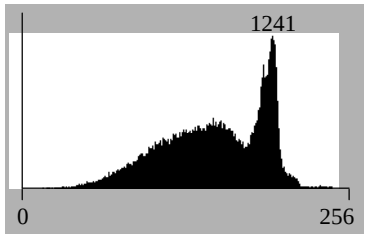
- Curve is called the cumulative distribution function

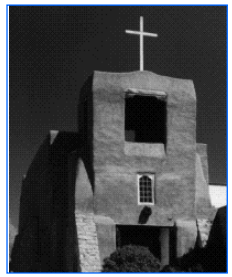
$$Q(i) = \sum_{k=0}^{i} P(k)$$

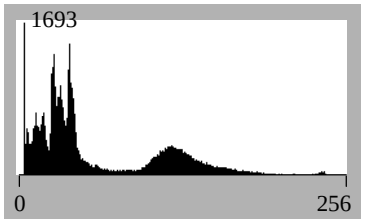
$$CH(i) = \sum_{k=0}^{i} H(k)$$



Examples

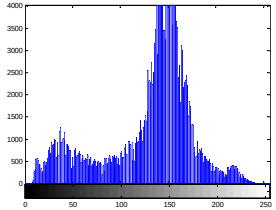






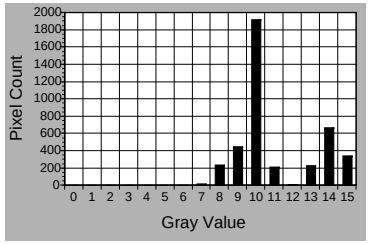
Histogram Equalization

- Image histograms consist of peaks, valleys, and low plains
- Peaks = many pixels concentrated in a few grey levels
- Plains = small number of pixels distributed over a wider range of grey levels

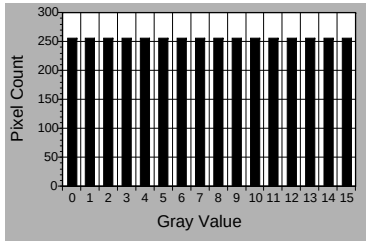


Histogram Equalization

- The goal is to modify the gray levels of an image so that the histogram of the modified image is flat (!!!?).
- Remark: only possible with continuous data
  In practice, the histogram might not become totally flat !
- Has a tendency to enhance contrast.



Original histogram.
Note low utilization of small gray values



Desired histogram

Brute Force

Gray Scale	Actual Count	Desired Count	How to get it
0	0	256	22 from 7, 234 from 8
1	0	256	1 from 8, 255 from 9
2	0	256	195 from 9, 61 from 10
3	0	256	256 from 10
4	0	256	256 from 10
5	0	256	256 from 10
6	0	256	256 from 10
7	22	256	256 from 10
8	235	256	256 from 10
9	450	256	256 from 10
10	1920	256	67 from 10, 189 from 11
11	212	256	23 from 11, 10 from 12, 223 from 13
12	10	256	10 from 13, 246 from 14
13	233	256	256 from 14
14	672	256	170 from 14, 86 from 15
15	342	256	256 from 15
Sum	4096	4096	

How are the gray levels in the original image changed to produce the enhanced image?

Method 1. Choose points randomly.

Method 2. Choice depends on the gray levels of their neighboring points.

Computationally expensive.

Approximations.

Histogram Equalization

- Mapping from one set of grey-levels, I, to a new set, O.
- Ideally, the number of pixels, N_p, occupying each grey level should be:

$$N_p = \frac{M*N}{G}$$

M, N - image size
G - no. gray levels

- To approximate this, apply the transform

$$i = MAX\left[\ 0,\ round\left\{\frac{CH(j)}{N_p}\right\} - 1\ \right]$$

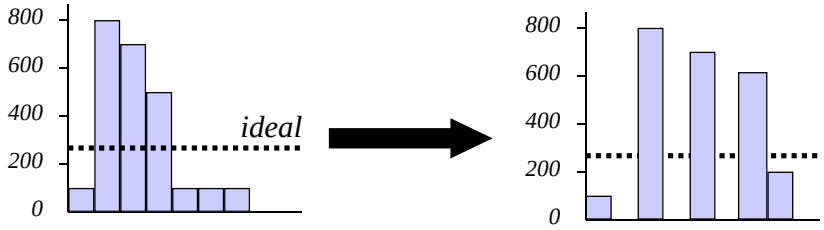
  - Where CH is the cumulative histogram (see next slide)
  - j is the gray value in the source image
  - i is the gray value in the equalized image

Example

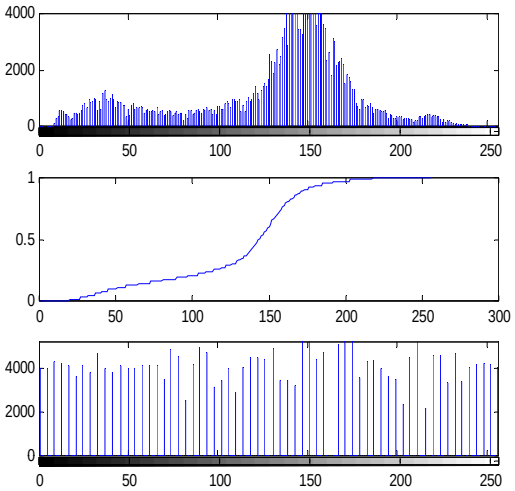
G=8
MxN=2400
N_p=300

$$CH(j) = \sum_{i=0}^{j} H(i)$$

j	*H(j)*	*CH(j)*	*i*
0	100	100	0
1	800	900	2
2	700	1600	4
3	500	2100	6
4	100	2200	6
5	100	2300	7
6	100	2400	7
7	0	2400	7



Example



Histogram Equalization Algorithm

- For an MxN image of G gray levels, say 0-255
  - Create image histogram
  - For cumulative image histogram H_c
- Set

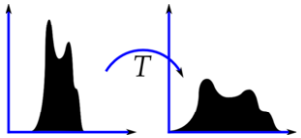
$$T(p) = round\left(\frac{G-1}{M*N}\ H_c(p)\right)$$

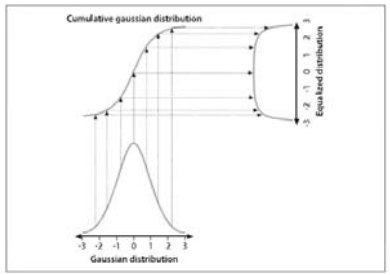
- Rescan input image and write new output image by setting

$$g_q = T(g_p)$$

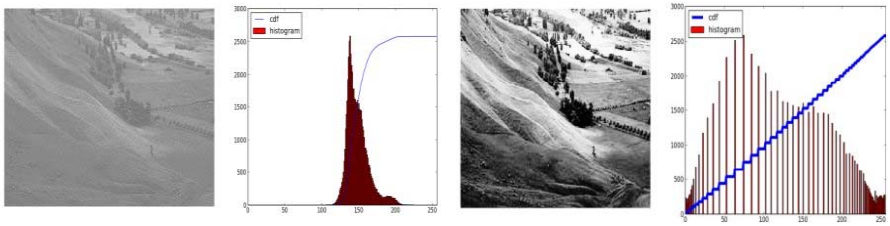
from: M. Sonka et. al., "Image Processing, Analysis, and Machine Vision"

Histogram Equalization Algorithm





Using the cumulative density function to equalize a Gaussian distribution



source: http://docs.opencv.org/

Comparison

Original | Histogram stretching | Histogram equalization

- The contrast enhancement is better after the histogram equalization, which more easily detects structures located in the shade.
- In fact any strongly represented gray-level is stretched while any weakly represented graylevel is merged with other close levels

Comparison

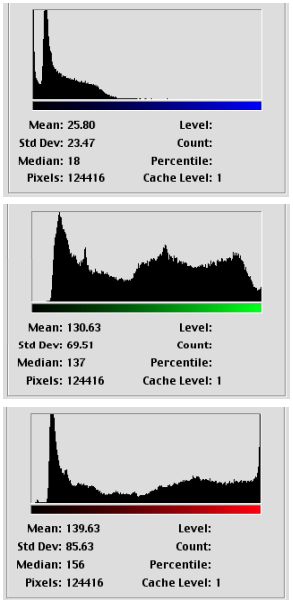
Original

γ>1

Histogram equalization

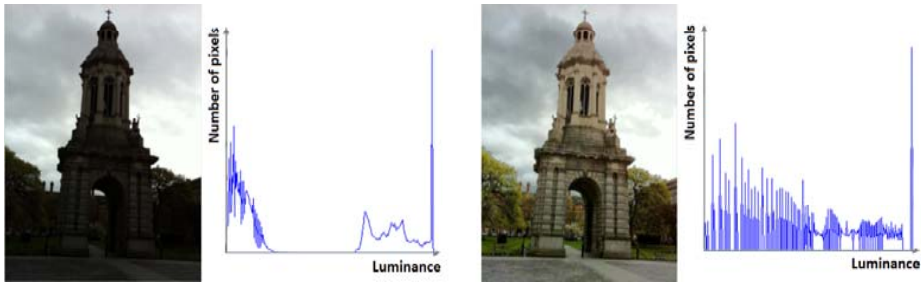
Color Histograms

http://docs.opencv.org/doc/tutorials/imgproc/histograms/histogram_calculation/histogram_calculation.html



Equalization of color images

- In color images,
  only the luminance channel is usually equalized
  as otherwise the colors can become distorted.



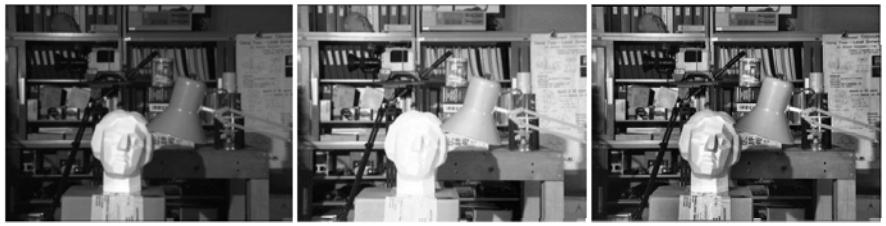
source: *A Practical Introduction to Computer Vision with OpenCV* , by Kenneth Dawson-Howe

Observations

- Acquisition process degrades image
- Brightness and contrast enhancement implemented by pixel operations
- No one algorithm universally useful
- $\gamma > 1$ enhances contrast in bright images
- $\gamma < 1$ enhances contrast in dark images
- Transfer function for histogram equalization proportional to cumulative histogram
- It is essential a process of trial-and-error to determine whether a particular type of images will benefit from histogram transformation operations.

CLAHE

- CLAHE – Contrast Limited Adaptive Histogram Equalization
  - Considering the global contrast of the image is not a good idea, in some cases ...
    (look at the face of the statue, in the middle image)
  - For some images,
    it might be preferable to apply different kinds of equalization in different regions.
  - Instead of computing a single curve, the image is divided into
    MxM pixel non-overlapped sub-blocks and
    separate histogram equalization is performed in each sub-block.
  - To avoid blocking artifacts (i.e., intensity discontinuities at block boundaries) I
    n the resulting image,
    the equalization functions are smoothly interpolated as we move between blocks.
  - This technique is known as adaptive histogram equalization (AHE) and
    its contrast limited (gain-limited) version is known as CLAHE.



Original image | After global equalization (notice the effect on the face of the statue) | After CLAHE

source: Szeliski's book + http://docs.opencv.org/

Noise reduction

Noise Reduction

- What is noise?
- How is noise reduction performed?
  - Noise reduction from first principles
  - Neighbourhood operators
    - linear filters (low pass, ~~high pass~~)
    - non-linear filters (median)

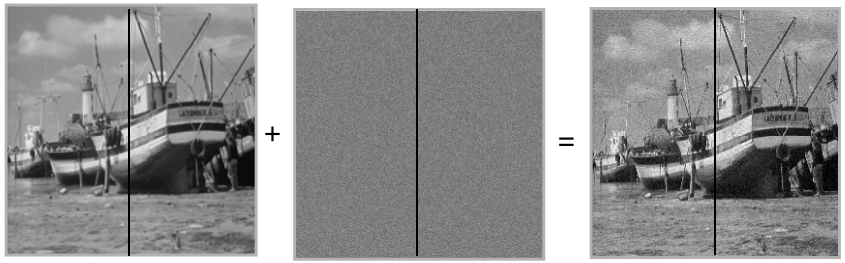
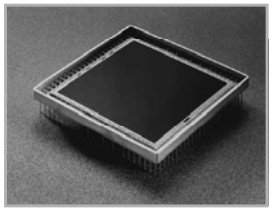
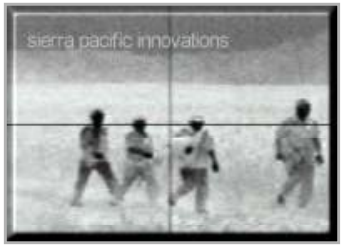


image + noise = 'grainy' image

Sources of Noise

- Sources of noise = CCD chip.
- Electronic signal fluctuations in detector.
- Caused by thermal energy.
- Worse for infra-red sensors.
- Electronics
- Transmission (analog)



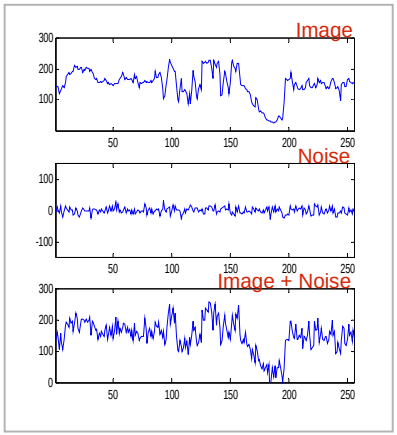


Radiation from the long wavelength IR band is used in most infrared imaging applications

Noise

- Plot of image brightness.
- Noise models:
  - additive
    - *f(i,j) = g(i,j) + v(i,j)*
      - Gaussian
      - salt and pepper
  - multiplicative
    - *f(i,j) = g(i,j) + g(i,j).v(i,j)*

  where
  - *f(i,j) - acquired signal (noisy)*
  - *g(i,j) - uncorrupted signal*
  - *v(i,j) – noise component*
- Noise fluctuations are rapid
  - high frequency.



Noise – salt & pepper

- Impulse noise
- Noise is maximum or minimum values

source: *A Practical Introduction to Computer Vision with OpenCV*, by Kenneth Dawson-Howe

Noise – Gaussian

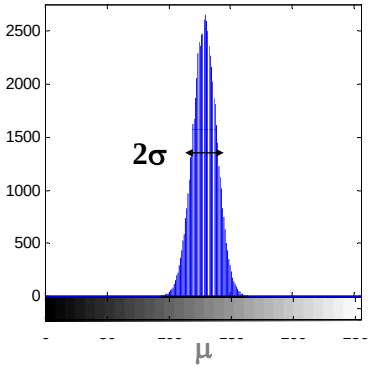
- Good approximation to real noise
- Distribution is Gaussian (mean & standard deviation)

source: *A Practical Introduction to Computer Vision with OpenCV*, by Kenneth Dawson-Howe

Noise Model

- Plot noise histogram
- Typical noise distribution is normal or Gaussian
- Mean(noise) $\mu = 0$
- Standard deviation σ

$$\eta(x) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma}\right)$$



Noise

- Noise varies above and below uncorrupted image.

Image

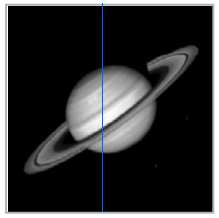
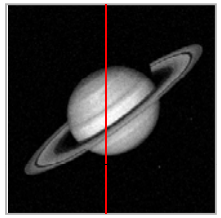
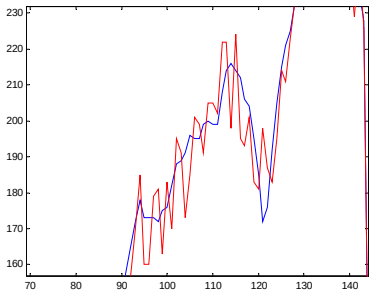


Image + Noise



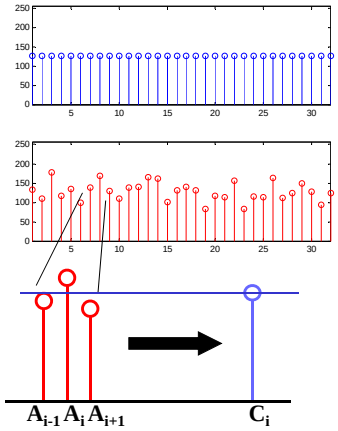


Noise reduction

- Removing (?) or reducing noise…
- Linear smoothing transformations
  - frame averaging (=> acquire several framesof static scene)
  - local averaging
  - Gaussian smoothing
- Non-linear transformations
  - median filter
  - rotating mask

Noise Reduction - 1

- How do we reduce noise?
- Consider a uniform 1-d image and add noise.
- Focus on a pixel neighbourhood.
- Averaging the three values should result in a value closer to original uncorrupted data
- Especially if Gaussian mean is zero

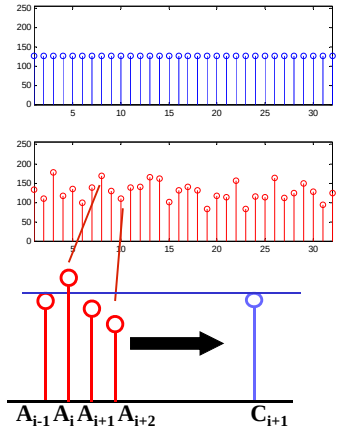


Introduction to Computer Vision

Noise Reduction - 1

- Averaging 'smoothes' the noise fluctuations.
- Consider the next pixel A_{i+1}
- Repeat for remainder of pixels.

$$C_{i+1} = \frac{A_i + A_{i+1} + A_{i+2}}{3}$$



Introduction to Computer Vision

Neighborhood Operations

- All pixels can be averaged by convolving 1-d image A with mask B to give enhanced image C.
- Weights of B must equal one when added together.
  - Why?

C = A * B

B = [B1 B2 B3]

$C_i = A_{i-1} \cdot B_1 + A_i \cdot B_2 + A_{i+1} \cdot B_3$

B = $\frac{1}{3}$ [1 1 1]

$$C_i = \frac{A_{i-1} + A_i + A_{i+1}}{3}$$

Introduction to Computer Vision

2D Analog of 1D Convolution

- Consider the problem of blurring a 2D image
- Here's the image and a blurred version:

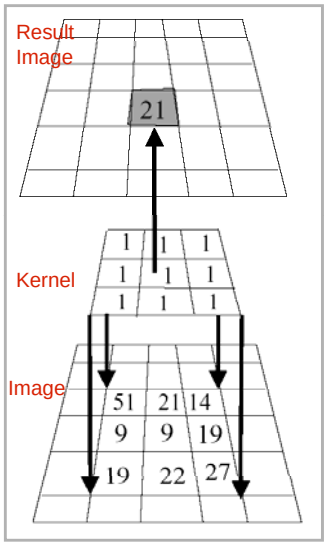
- How would we do this?
  - What's the 2D analog of 1D convolution?

2D Blurring Kernel

- C = A * B
- B becomes

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

Example



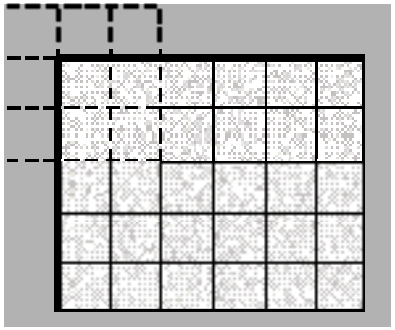
- Kernel is aligned with pixel in image, multiplicative sum is computed, normalized, and stored in result image.
- Process is repeated across image.
- What happens when kernel is near edge of input image?

$$\begin{bmatrix} 1*51 + 1*21 + 1*14 + \\ 1*8 + 1*9 + 1*19 + \\ 1*19 + 1*22 + 1*27 \end{bmatrix} 1/9 = 21$$

Border Problem

- missing samples are zero
- missing samples are gray
- copying last lines
- reflected indexing (mirror)
- circular indexing (periodic)
- reduce size of resulting image

- OpenCV
  allows some control on how to do this;
  see CopyMakeBorder(): copies the source 2D array into the interior of the destination array and makes a border of the specified type around the copied area



Cross-correlation

- Let's write this down as an equation. Assume the averaging window is (2k+1)x(2k+1):
- We can generalize $G[i,j] = \frac{1}{(2k+1)^2}\sum_{u=-k}^{k}\sum_{v=-k}^{k} F[i+u, j+v]$ ixels:
- This is called a **cross-correlation** operation and written:
- H is called the "filter," "kernel," or "mask."
- The above allows
  When you implem
  H[u+k,v+k] instead $G[i,j] = \sum_{u=-k}^{k}\sum_{v=-k}^{k} H[u,v]F[i+u, j+v]$

$$G = H \otimes F$$

Convolution

- A **convolution** operation is a cross-correlation where the filter is flipped both horizontally and vertically before being applied to the image:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i-u,j-v]$$

- It is written:

$$G = H \star F$$

- Suppose H is a Gaussian or mean kernel. How does convolution differ from cross-correlation?

Convolution

- Extremely important concept in computer vision, image processing, signal processing, etc.
- Lots of related mathematics (we won't do)
- General idea: reduce a filtering operation to the repeated application of a mask (or filter kernel) to the image
  - Kernel can be thought of as an NxN image
  - N is usually odd so kernel has a central pixel
- In practice
  - (flip kernel)
  - Align kernel center pixel with an image pixel
  - Pointwise multiply each kernel pixel value with corresponding image pixel value and add results
  - Resulting sum is normalized by kernel weight
  - Result is the value of the pixel centered on the kernel

Convolution

Image Data:

10	12	40	16	19	10
14	22	52	10	55	41
10	14	51	21	14	10
32	22	9	**9**	19	14
41	18	9	22	27	11
10	7	8	8	4	5

Mask:

1	1	1	1
0	1	1	1
-1	1	1	1
	-1	*0*	*1*

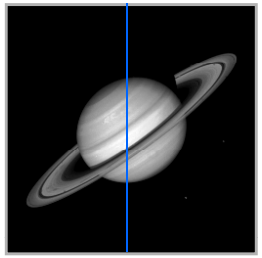
$$S = \sum_{k=-1}^{1} \sum_{m=-1}^{1} M(k,m)$$

$$I_f(i,j) = \frac{1}{S} \sum_{k=-1}^{1} \sum_{m=-1}^{1} I(i+k,j+m) * M(k,m) = I \circledast M$$

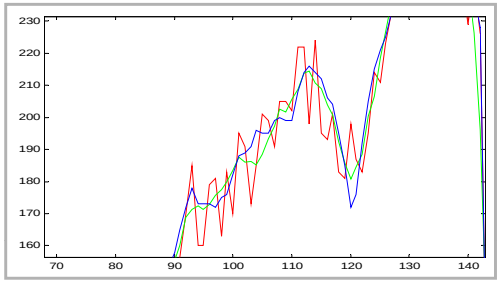
Properties of Convolution

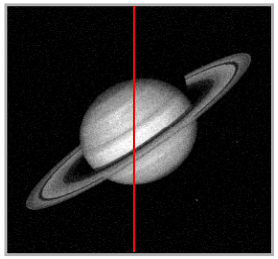
- Commutative: $f_1(t) * f_2(t) = f_2(t) * f_1(t)$
- Distributive: $f_1(t) * [f_2(t) + f_3(t)] = f_1(t) * f_2(t) + f_1(t) * f_3(t)$
- Associative: $f_1(t) * [f_2(t) * f_3(t)] = [f_1(t) * f_2(t)] * f_3(t)$
- Shift: if $f_1(t) * f_2(t) = c(t)$, then
  - $f_1(t) * f_2(t-T) = f_1(t-T) * f_2(t) = c(t-T)$
- Convolution with impulse: $f(t) * \delta(t) = f(t)$
- Convolution with shifted impulse: $f(t) * \delta(t-T) = f(t-T)$

Noise Reduction - 1



Uncorrupted Image





Uncorrupted Image + Noise

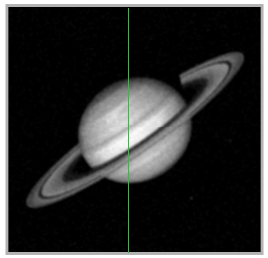


Image + Noise - Blurred

Noise Reduction - 1

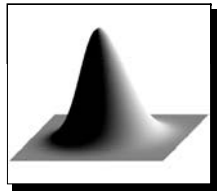
- Technique relies on high frequency noise fluctuations being 'blocked' by filter.
  Hence, low-pass filter.
- Fine detail in image may also be smoothed.
- Balance between
  keeping image fine detail and
  reducing noise.

Noise Reduction - 1

- Saturn image
  (*previous slide*)
  has coarse detail
- Boat image contains
  fine detail
- Noise reduced but
  fine detail also smoothed

Gaussian smoothing

- Smoothing operator should be
  - 'tunable' in what it leaves behind
  - smooth and localized in image space.
- One operator which satisfies
  these two constraints is the Gaussian



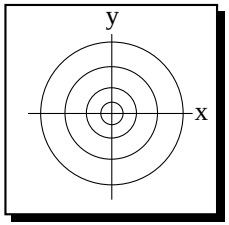
- *OpenCV: Smooth()*

2D Gaussian Distribution

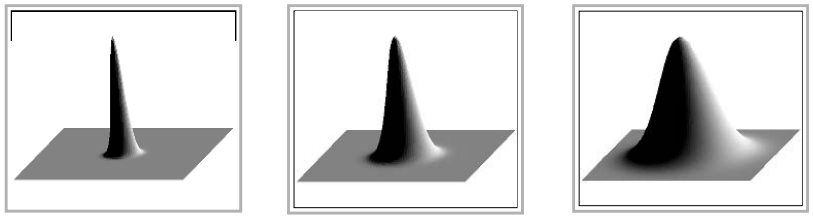
- The two-dimensional Gaussian distribution is defined by:

$$G(x,y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\left[\frac{(x^2+y^2)}{2\sigma^2}\right]}$$

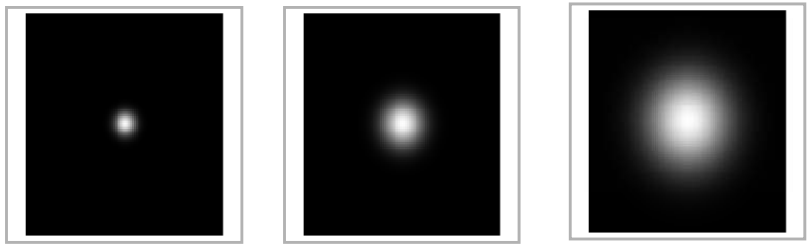
- From this distribution, can generate smoothing masks whose width depends upon σ:



σ Defines Kernel "width"



$\sigma^2 = .25$ $\sigma^2 = 1.0$ $\sigma^2 = 4.0$



Creating Gaussian Kernels

- The mask weights are evaluated from the Gaussian distribution:

$$W(i,j) = k * \exp\left(-\frac{i^2 + j^2}{2\sigma^2}\right)$$

- This can be rewritten as:

$$\frac{W(i,j)}{k} = \exp\left(-\frac{i^2 + j^2}{2\sigma^2}\right)$$

- This can now be evaluated over a window of size nxn to obtain a kernel in which the (0,0) value is 1.
- k is a scaling constant

Example

- Choose $\sigma^2 = 2$ and n = 7, then:

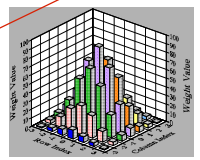
i \ j	-3	-2	-1	0	1	2	3
-3	.011	.039	.082	.105	.082	.039	.011
-2	.039	.135	.287	.368	.287	.135	.039
-1	.082	.287	.606	.779	.606	.287	.082
0	.105	.039	.779	1.000	.779	.368	.105
1	.082	.287	.606	.779	.606	.287	.082
2	.039	.135	.287	.368	.287	.135	.039
3	.011	.039	.082	.105	.082	.039	.011

$$\frac{W(1,2)}{k} = \exp\left(-\frac{1^2+2^2}{2*2}\right)$$

To make this value 1, choose k=91

1	4	7	10	7	4	1
4	12	26	33	26	12	4
7	26	55	71	55	26	7
10	33	71	91	71	33	10
7	26	55	71	55	26	7
4	12	26	33	26	12	4
1	4	7	10	7	4	1

7x7 Gaussian filter



Kernel Application

7x7 Gaussian kernel

15x15 Gaussian kernel

Why Gaussian for Smoothing

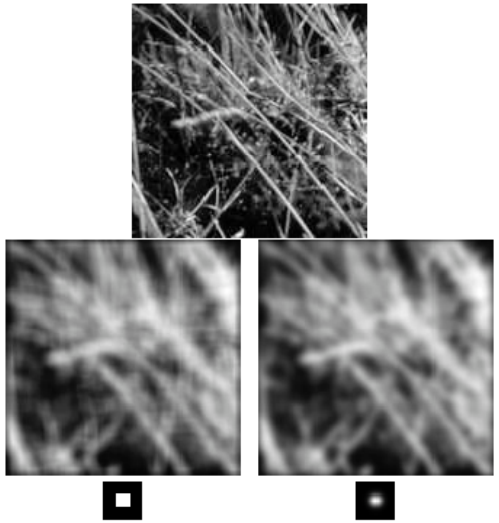
- Gaussian is not the only choice, but it has a number of important properties
  - If we convolve a Gaussian with another Gaussian, the result is a Gaussian
    - This is called linear scale space

$$G_{\sigma_1} * G_{\sigma_2} = G_{\sqrt{\sigma_1^2+\sigma_2^2}}.$$

  - Efficiency: separable

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$$
$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(x^2)}{2\sigma^2}\right)\right) \times \left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(y^2)}{2\sigma^2}\right)\right).$$

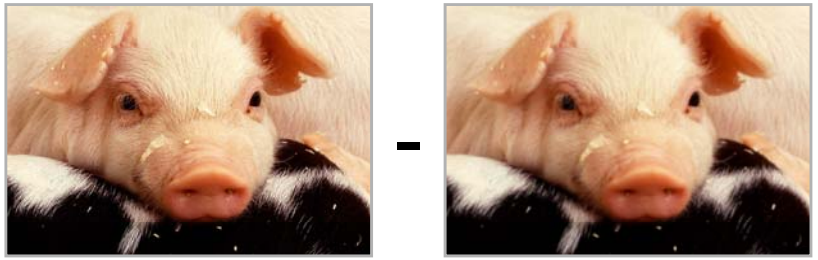
Mean vs. Gaussian filtering

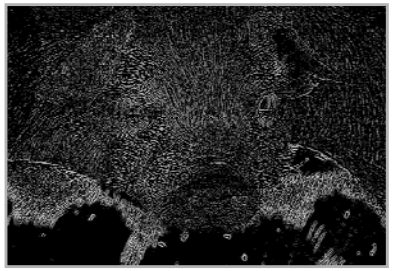


hmmmmm…..

Image

Blurred Image





Noise Reduction - 2: Median Filter

- Compute median of points in neighborhood
- Nonlinear filter
  - example of a larger class of filters named "rank-filters" (ex: min, median, max)
- Has a tendency to blur detail less than averaging
- Works very well for 'shot' or 'salt and pepper' noise

Original Low-pass Median

Mean vs. Median

- Mean
  - Linear
  - Signal frequencies shared with noise are lost, resulting in blurring.
  - Impulsive noise is diffused but not removed.
  - It spreads the noise, resulting in blurring.
- Median
  - Non-linear
  - Does not spread the noise
  - Can remove spike noise
  - Preserves (some) edges
  - Expensive to run

Filtering

- Filtering is perhaps the most fundamental operation of image processing and computer vision.
- In the broadest sense of the term "filtering," the value of the filtered image at a given location is a function of the values of the input image in a small neighborhood of the same location.
- In particular, Gaussian low-pass filtering computes a weighted average of pixel values in the neighborhood, in which, the weights decrease with distance from the neighborhood center. Although formal and quantitative explanations of this weight fall-off can be given, the intuition is that images typically vary slowly over space, so near pixels are likely to have similar values, and it is therefore appropriate to average them together. The noise values that corrupt these nearby pixels are mutually less correlated than the signal values, so noise is averaged away while signal is preserved.
- The assumption of slow spatial variations fails at edges, which are consequently blurred by low-pass filtering. Many efforts have been devoted to reducing this undesired effect.

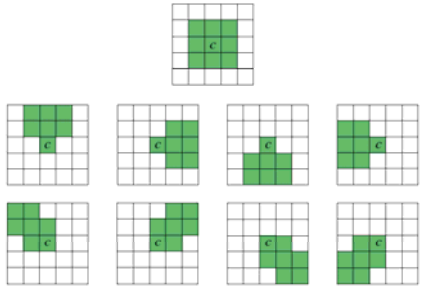
Edge Preserving Smoothing

- Smooth (average, blur) an image without disturbing
  - sharpness or
  - position

  of the edges
- Nagoa-Maysuyama filter
- Kuwahara filter
- Anisotropic diffusion filter (Perona & Malik, ...)
- Bilateral filtering
- ...

Nagao-Matsuyama filter

- Calculate the variance within nine subwindows of a 5x5 moving window
- Output value is the **mean** of the **subwindow with the lowest variance**
- Nine subwindows used:



Kuwahara filter

- Principle:
  - divide filter mask into four regions (a, b, c, d).
  - in each compute the mean brightness and the variance
  - the output value of the center pixel (abcd) in the window is the **mean** value of that **region** that has the **smallest variance**.

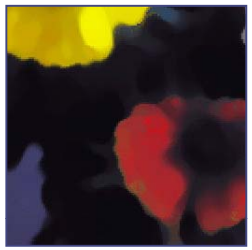
a	a	ab	b	b
a	a	ab	b	b
ac	ac	abcd	bd	bd
c	c	cd	d	d
c	c	cd	d	d

Kuwahara Filter Example

Original

Median (1 iteration)

Kuwahara

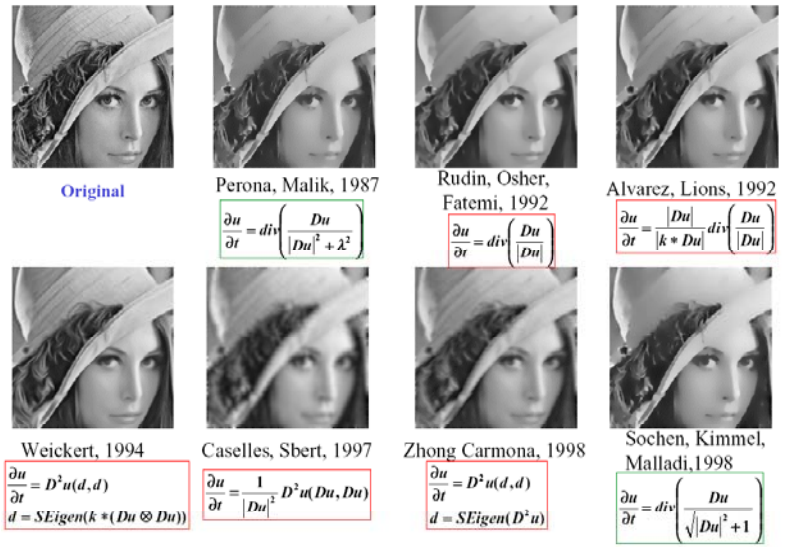


Median (10 iterations)

Anisotropic Diffusion Filtering

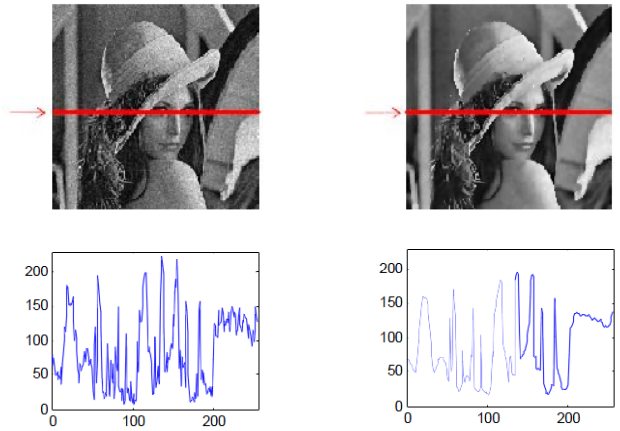
- The anisotropic diffusion is a nonlinear filter that smoothes the intraregions of an image without blurring the strong edges
  - Encourages the **smoothing in homogeneous regions in preference to** smoothing across the boundaries.
- Based on the solution of a partial differential equation, inspired in heat diffusion equation
  - *The algorithm results from the discretization of the non-linear diffusion equation.*
  - *The derivatives are approximated by differences.*
- Diffusion methods average over extended regions by solving partial differential equations, and are therefore inherently iterative. Iteration may raise issues of stability and, depending on the computational architecture, efficiency.
- This algorithm is also used to detect the edges

Anisotropic Diffusion Filtering



Note: Original Perona-Malik diffusion process is NOT anisotropic, although they erroneously said it was.

Anisotropic Diffusion Filtering



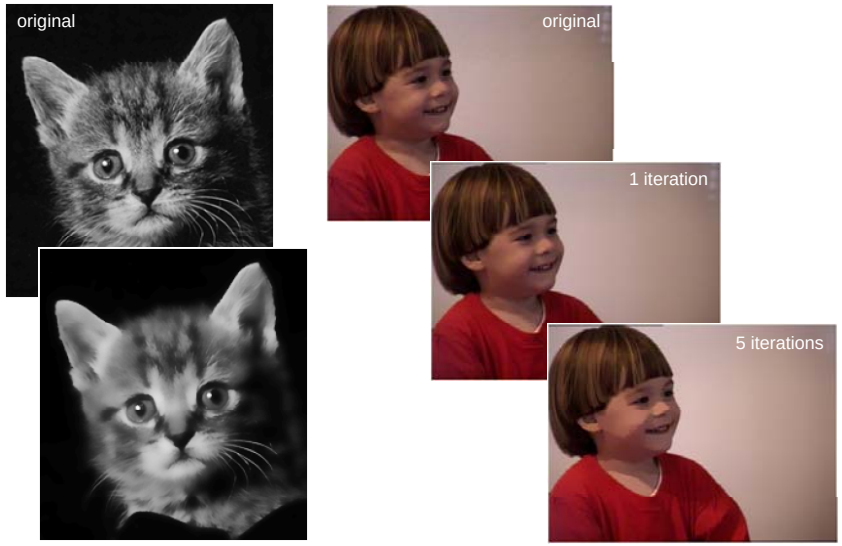
Example: Perona-Malik

- In general:
  - Anisotropic diffusion estimates an image from a noisy image
  - Useful for restoration, etc.
  - Only shown smoothing examples
  - *Read the literature*

Bilateral filtering

- A bilateral filter is an edge-preserving and noise reducing smoothing filter.
- Traditional filtering is **domain filtering**, and enforces closeness by weighing pixel values with coefficients that fall off with distance.
- Similarly, we define **range filtering**, which averages image values with weights that decay with dissimilarity.
- **Range filters**
  - are nonlinear because their weights depend on image intensity or color
  - preserve edges
  - by themselves, my distort an image's color map (see C.Tomasi at al., Bilateral Filtering for Gray and Color, ICCV 1998)
- **Bilateral filtering** combines range and domain filtering.
- The intensity value at each pixel in an image is replaced by a weighted average of intensity values from nearby pixels.
- This weight is based on a Gaussian distribution. The **weights** depend not only on Euclidean distance but also on the radiometric differences (e.g. color intensity).
- It preserves sharp edges by systematically looping through each pixel and attributing weights to the adjacent pixels accordingly.
- *OpenCV: Smooth()*

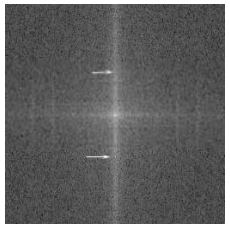
Bilateral filtering



Frequency domain filtering

- An image can be represented in the frequency domain, using the Fourier Transform (FT).
- The FT encodes the amplitude and phase of each frequency component.
- The values near the origin of the transformed space are called low-frequency components of the FT, and those distant from the origin are the high-frequency components.
- Convolution in the image domain corresponds to multiplication in the spatial frequency domain.
- Therefore, convolution with large filters, which would normally be expensive in the image domain, can be implemented efficiently using the Fast Fourier Transform (FFT).
- This is an important technique in many image processing applications.
- In machine vision, however, most algorithms are nonlinear or spatially varying, so the FFT cannot be used.

Frequency domain filtering



2D Fourier transform

After filtering