Digital Image Filtering

Rüdiger Gens

UNIVERSITY OF ALASKA FAIRBANKS

Outline

- digital image filtering
- filter types
  - low-pass, high-pass
  - directional, edge detection
  - morphological
  - texture
  - speckle reduction
- applications

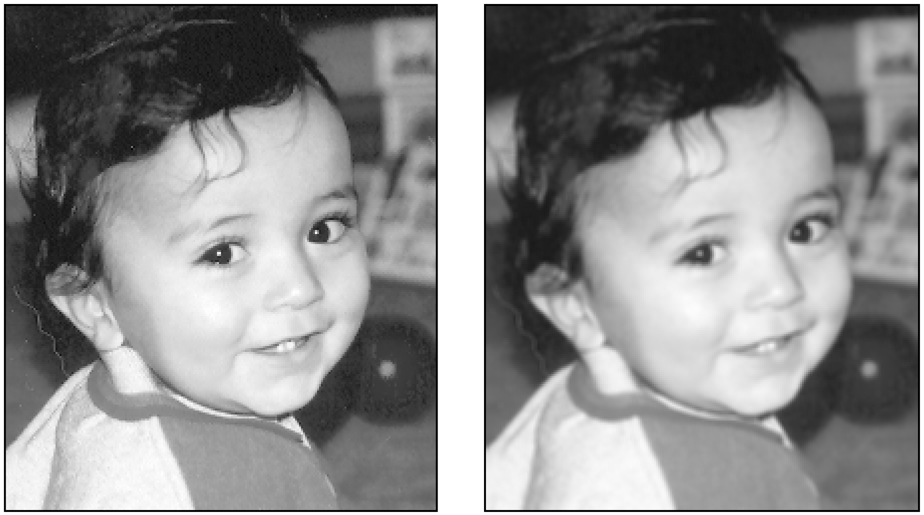
Digital image filtering

- sometimes referred to as convolution
- neighborhood operation
- filtering manipulates (removes, reduces, or enhances) one of the image components
  - low frequency (subtle change in pixel values)
  - high frequency (sudden change in pixel values)
  - noise / speckle
- uses a kernel (small image window) that is multiplied with each pixel in the image

Digital image filtering

- various methods of treating the edge pixels
  - zero values
  - input pixel values
  - input image boundary mirroring before filtering
- keeping the dynamic range of pixel values
  - normalizing the kernel
- dimension of kernels
  - mostly equal dimensional, sometimes linear
  - mostly odd number of kernel elements

Low-pass filter



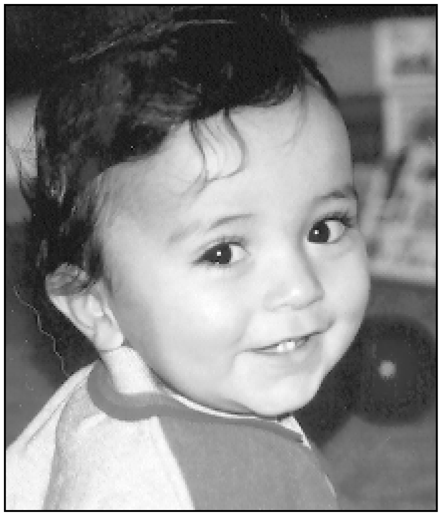
Original 3x3 low-pass

- used to suppress high-frequency variation
- used to suppress noise
- has smoothing effect

1	1	1
1	1	1
1	1	1

Kernel

Filter strength



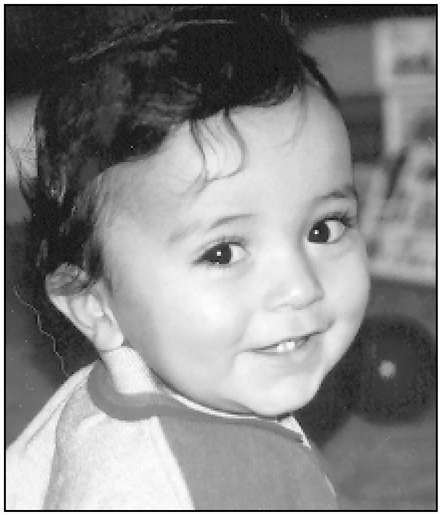
Original

3x3 low-pass

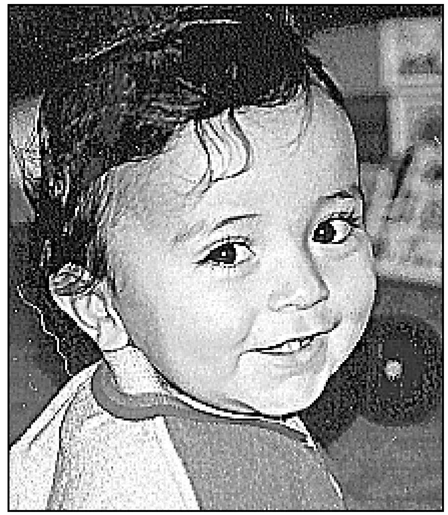
5x5 low-pass

- filtering strength related to kernel size and weighting

High-pass filter



Original



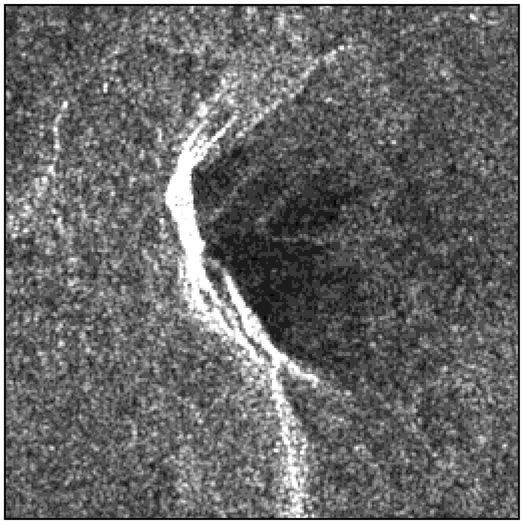
3x3 high-pass

-1	-1	-1
-1	9	-1
-1	-1	-1

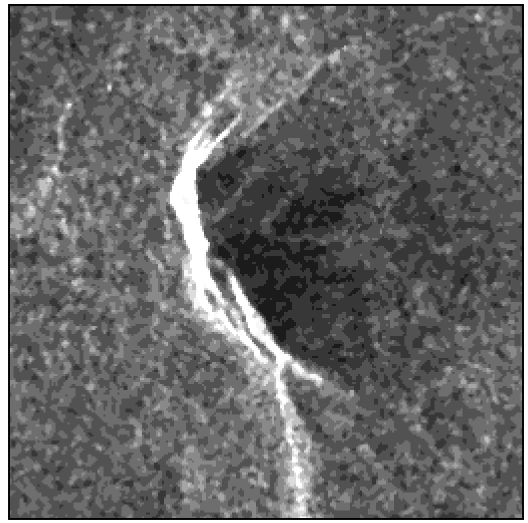
Kernel

- used to enhance high-frequency variations
- noise usually gets enhanced as well
- can enhance variations in certain directions

Median filter



Original

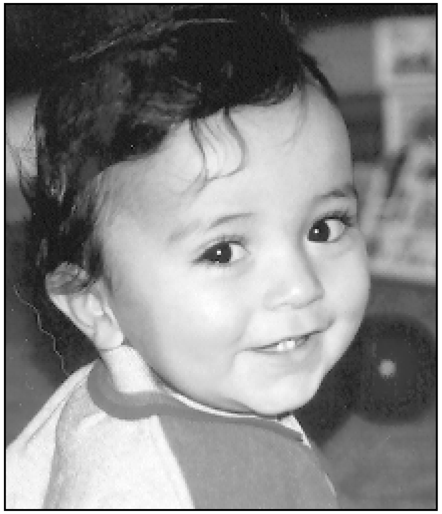


3x3 Median

- less blurring than low-pass filter
- less sensitive to extreme values

$$\text{pixel value} = \text{middle value in an ordered set of values}$$
$$= (3, 5, 6, 7, \mathbf{9}, 10, 11, 14, 25)$$

Directional filter



Original

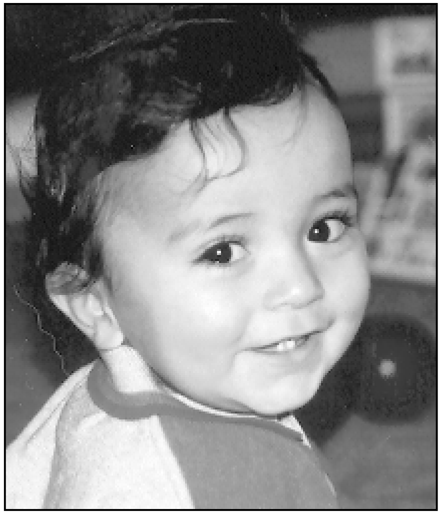
3x3 vertical edge

- enhances vertical lines and edges

-1	0	1
-2	0	2
-1	0	1

Kernel

Directional filter


Original

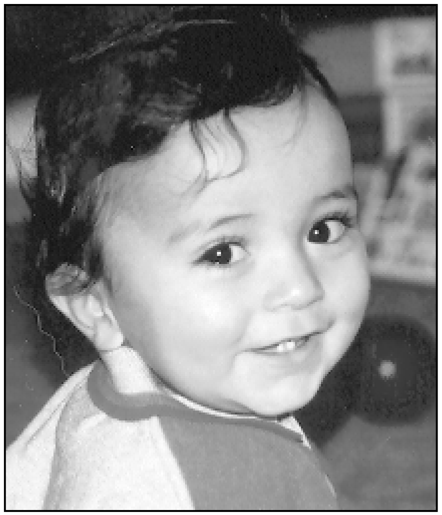

3x3 horizontal edge

- enhances horizontal lines and edges

1	2	1
0	0	0
-1	-2	1

Kernel

Edge detection – Sobel



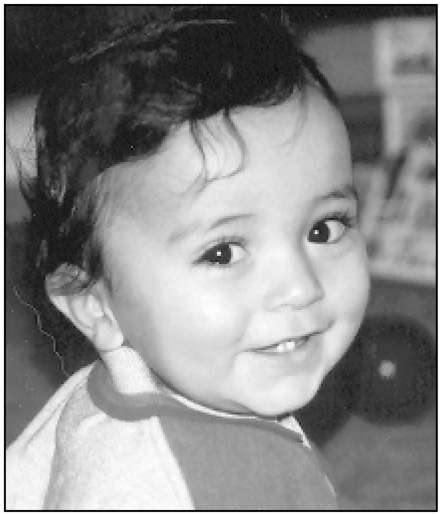
Original

3x3 Sobel

- uses horizontal and vertical direction filters
- calculates the magnitude of the two gradients
- limited use with noisy imagery

$$G_m = \sqrt{G_x^2 + G_y^2}$$

Edge detection – Laplace



Original

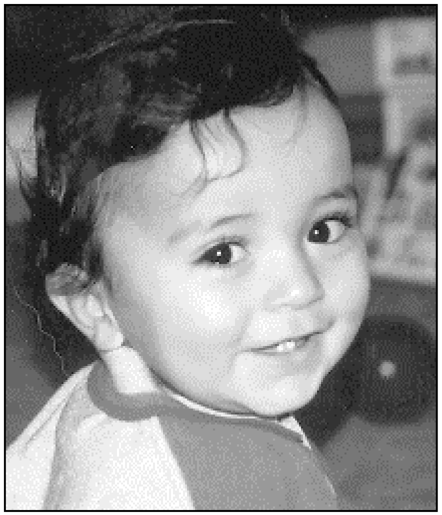
3x3 Laplace

- non-directional filter
- symmetric
- various implementations with different weighting strategies

-1	-1	-1
-1	8	-1
-1	-1	-1

Kernel

Morphological filter – Erosion

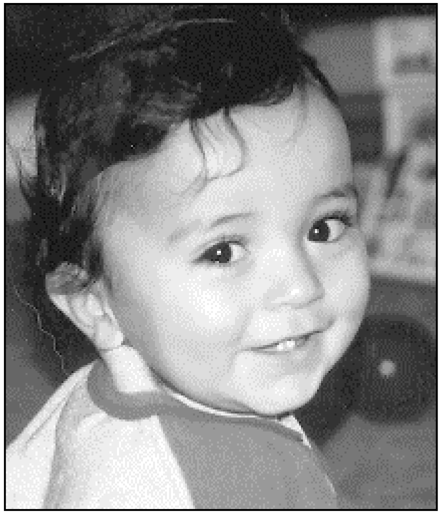


Original

3x3 Erosion

- minimum value within filter window
- reduces bright areas

Morphological filter – Dilation

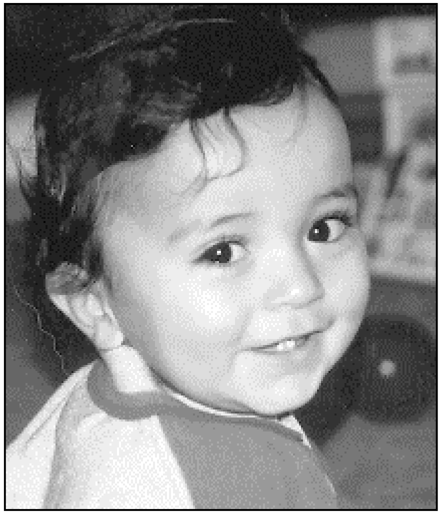


Original

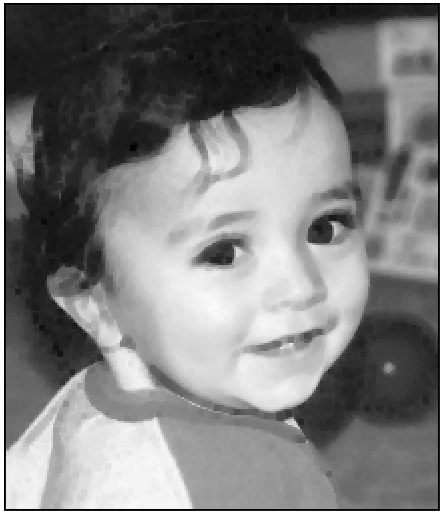
3x3 Dilation

- maximum value within filter window
- enhances bright areas

Morphological filter – Opening



Original

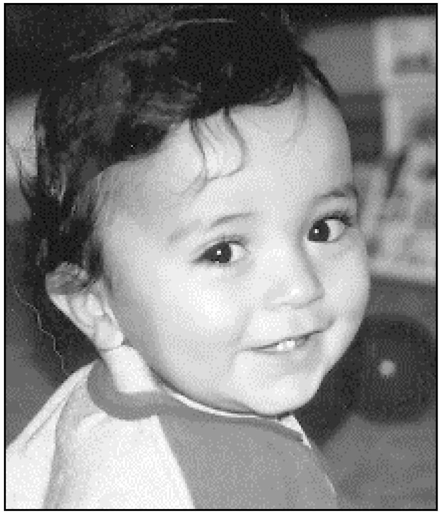


3x3 Erosion

3x3 Opening

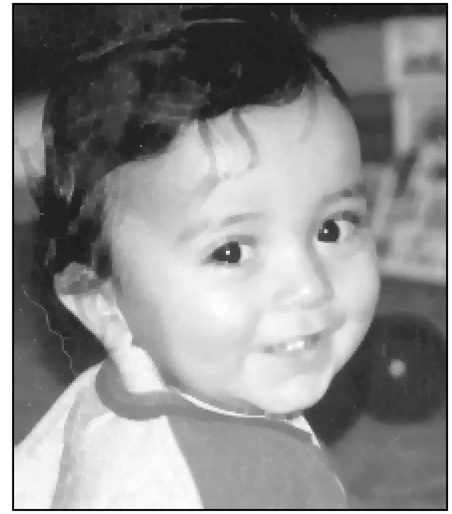
- erosion followed by dilation

Morphological filter – Closing



Original

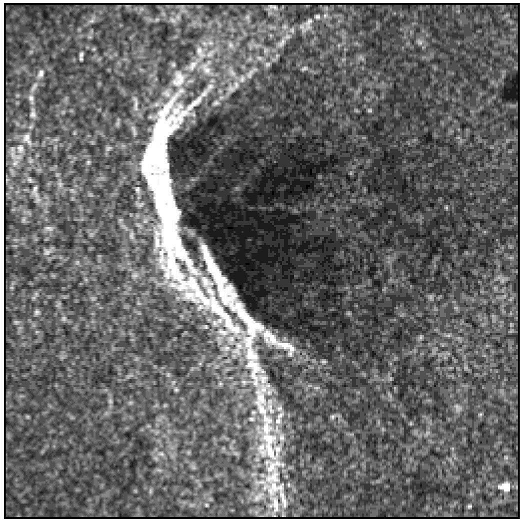
3x3 Dilation



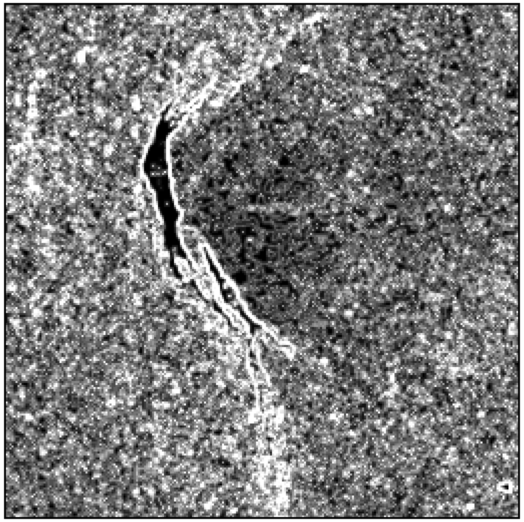
3x3 Closing

- dilation followed by erosion

Texture – Mean Euclidian distance



Original



3x3 Mean Euclidian distance

- first-order moment about the mean

$$\text{mean Euclidian distance} = \frac{\sqrt{\sum \left(p_{i,j} - p_c\right)^2}}{n-1}$$

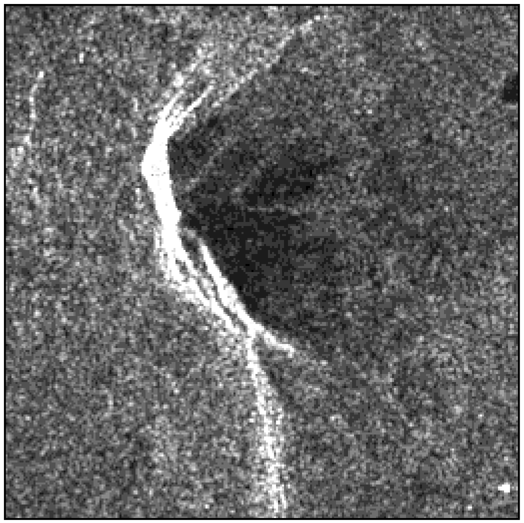
where

$p_{i,j}$ = pixel value at location (i, j) in the filter window

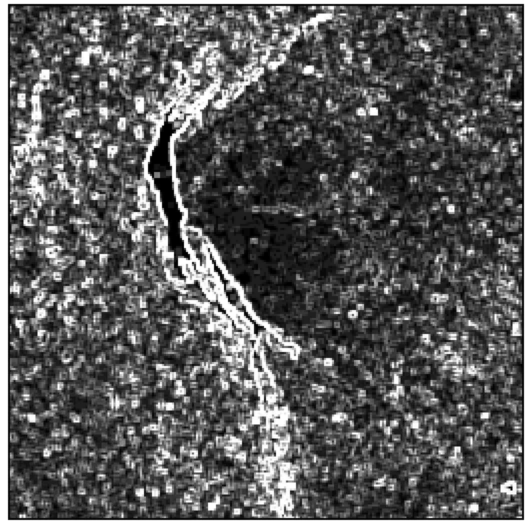
p_c = center pixel value

n = number of pixels in the filter window

Texture – Variance



Original



3x3 Variance

- second-order moment about the mean
- a measure of gray tone variance within the window

$$\text{variance} = \frac{\sum \left(p_{i,j} - M\right)^2}{n-1}$$

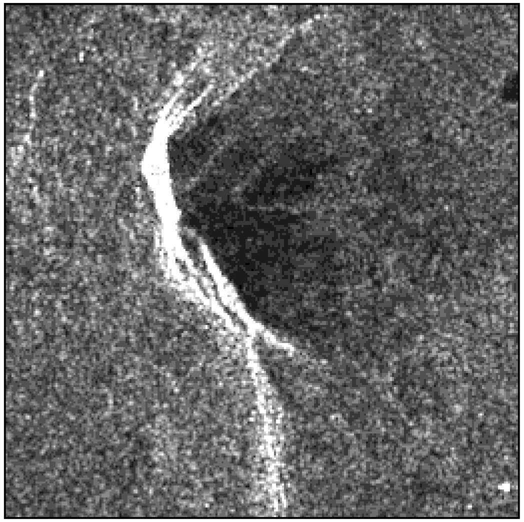
where

$p_{i,j}$ = pixel value at location (i, j) in the filter window

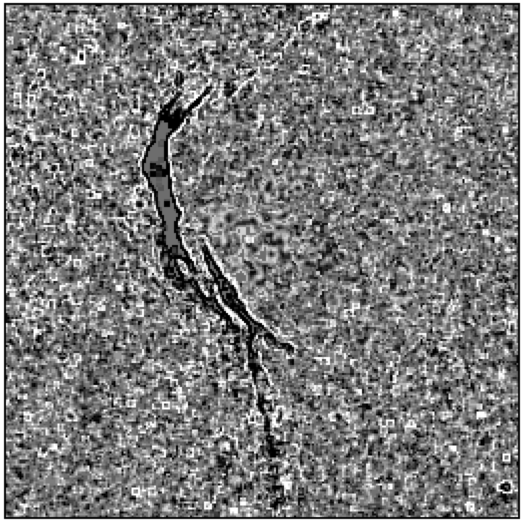
M = mean pixel value

n = number of pixels in the filter window

Texture – Skewness



Original



3x3 Skewness

- third order moment about the mean
- departure from symmetry about the mean gray level

$$\text{skew} = \frac{\left|\sum \left(p_{i,j} - M\right)^3\right|}{(n-1) * \sqrt{\text{var}^3}}$$

where

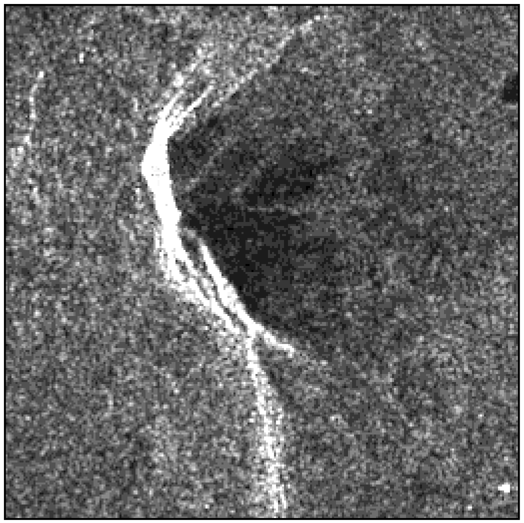
$p_{i,j}$ = pixel value at location (i, j) in the filter window

M = mean pixel value

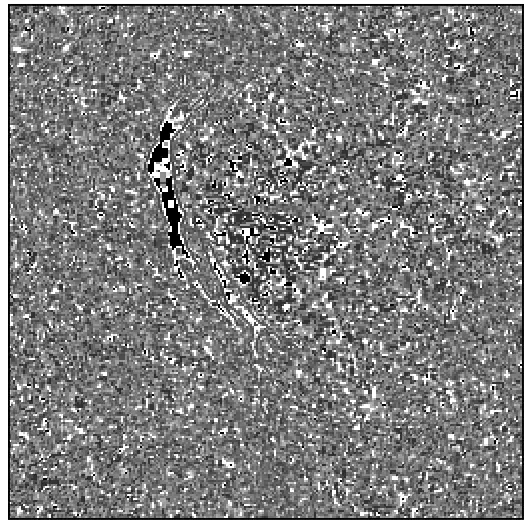
var = variance of pixel values

n = number of pixels in the filter window

Texture – Kurtosis



Original



3x3 Kurtosis

- fourth order moment about the mean
- measure of the spread of gray tones about the mean

$$\text{kurtosis} = \frac{\sum (p_{i,j} - M)^4}{(n-1) * \text{var}^2}$$

where

$p_{i,j}$ = pixel value at location (i, j) in the filter window

M = mean pixel value

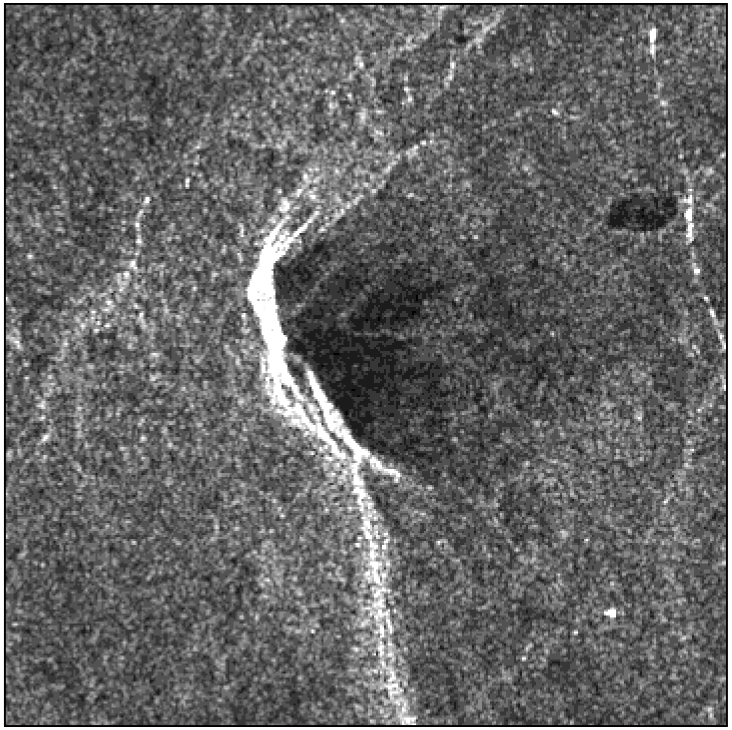
var = variance of pixel values

n = number of pixels in the filter window

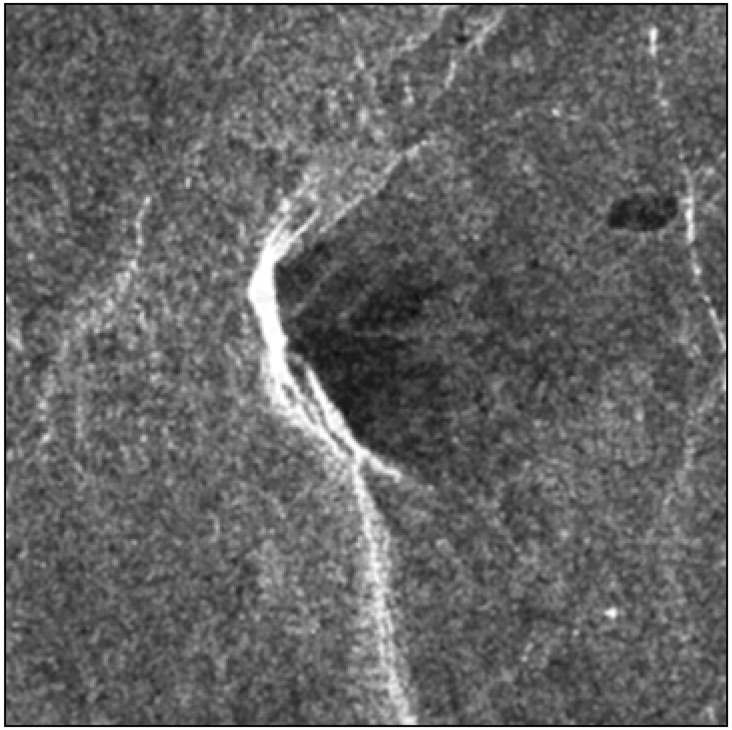
Speckle reduction filters

- mostly applicable to SAR imagery
  - accounts for number of looks
- use local statistics
  - mean value, standard deviation etc.
- variety of filters developed over the years
  - (enhanced) Lee
  - (enhanced) Frost
  - Kuan
  - Gamma MAP

Speckle – Lee



Original



3x3 Lee

Speckle – Lee

- kernel math

$$\text{pixel value} = CP * W + I * (1 - W)$$

where

N_{look} = number of looks

CP = center pixel value

I = mean pixel value

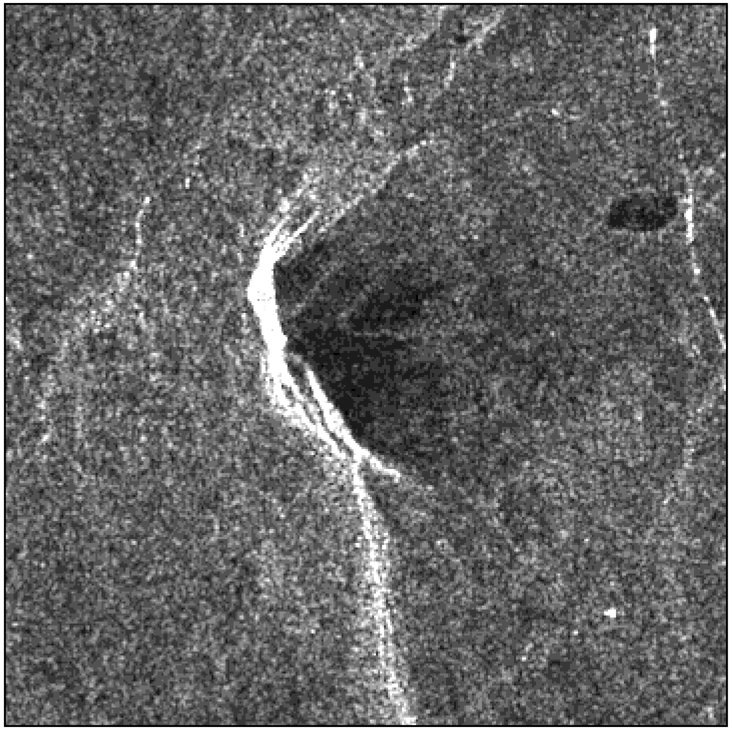
S = standard deviation of pixel values within filter window

$W = 1 - C_u^2 / C_i^2$

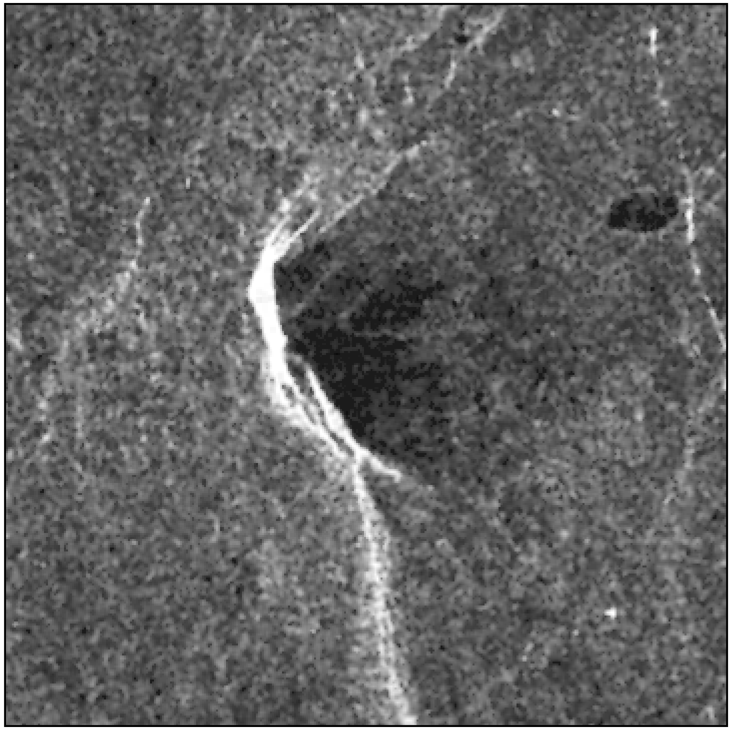
$C_u = \sqrt{1/N_{look}}$

$C_i = S/I$

Speckle – Enhanced Lee



Original



3x3 Enhanced Lee

Speckle – Enhance Lee

- kernel math

$$\text{pixel value} = \begin{cases} I & \text{for } C_i <= C_u \\ I * W + CP * (1 - W) & \text{for } C_u < C_i < C_{max} \\ CP & \text{for } C_i >= C_{max} \end{cases}$$

where

N_{look} = number of looks

CP = center pixel value

I = mean pixel value

S = standard deviation of pixel values within filter window

$DAMP$ = damping factor

$W = \exp^{-DAMP*((C_i - C_u)/(C_{max} - C_i))}$

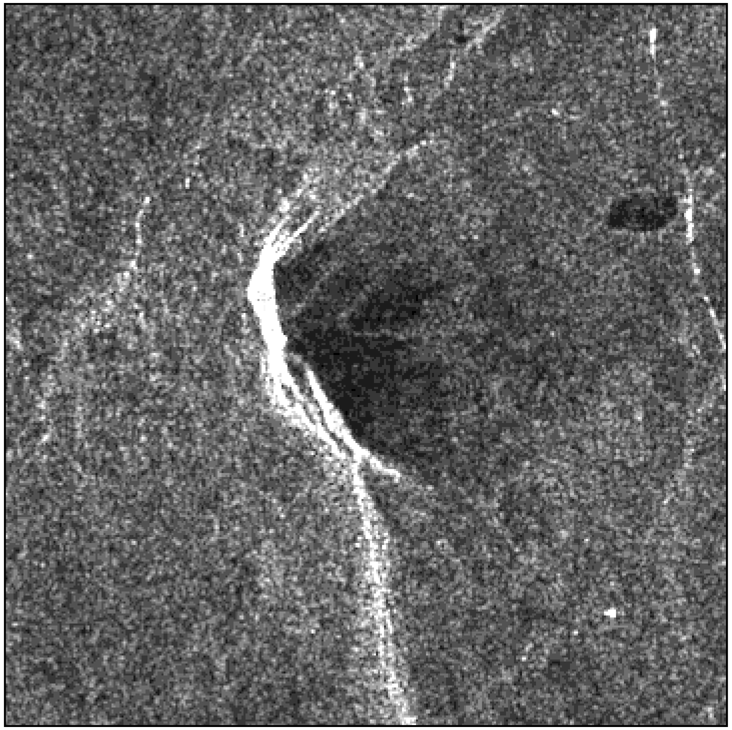
$C_u = \sqrt{1/N_{look}}$

$C_i = S/I$

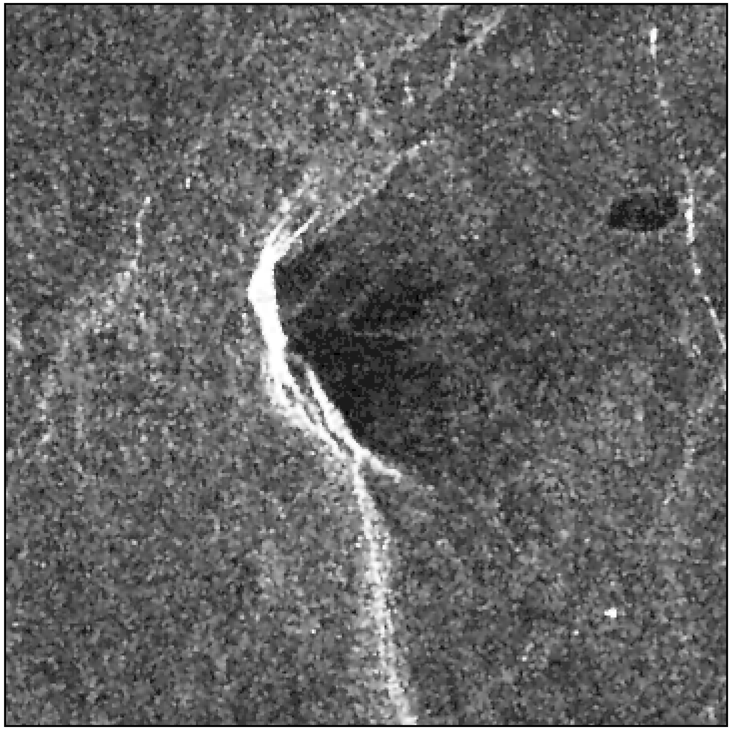
$C_{max} = \sqrt{1 + 2/N_{look}}$

Speckle – Frost

Digitall image filtering



Original



3x3 Frost

Speckle – Frost

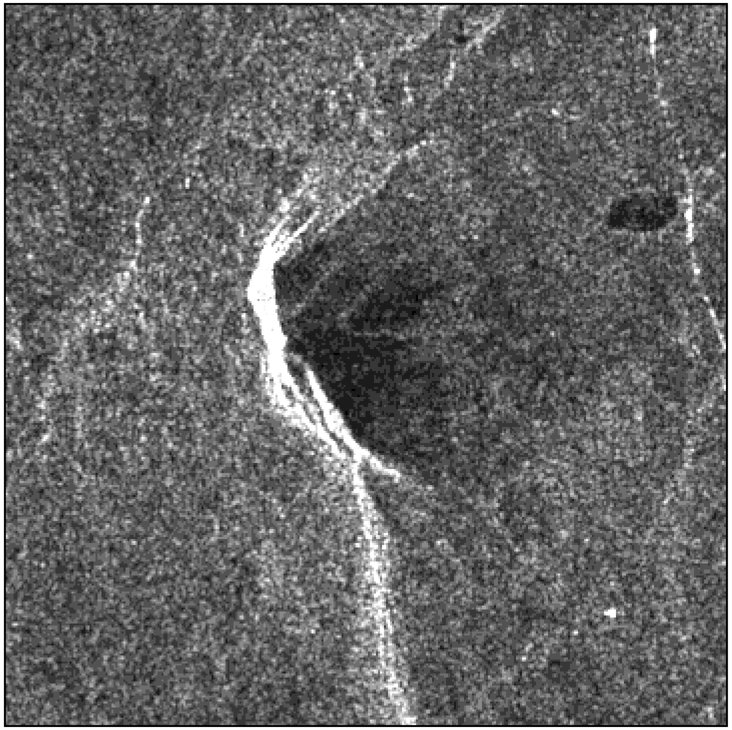
- Kernel math

$$\text{pixel value} = \sum_{i=1}^{m}\sum_{j=1}^{n}(i,j) * M_{i,j} \Big/ \sum_{i=1,j=1}^{m,n} \mathrm{M}_{\mathrm{i,j}}$$

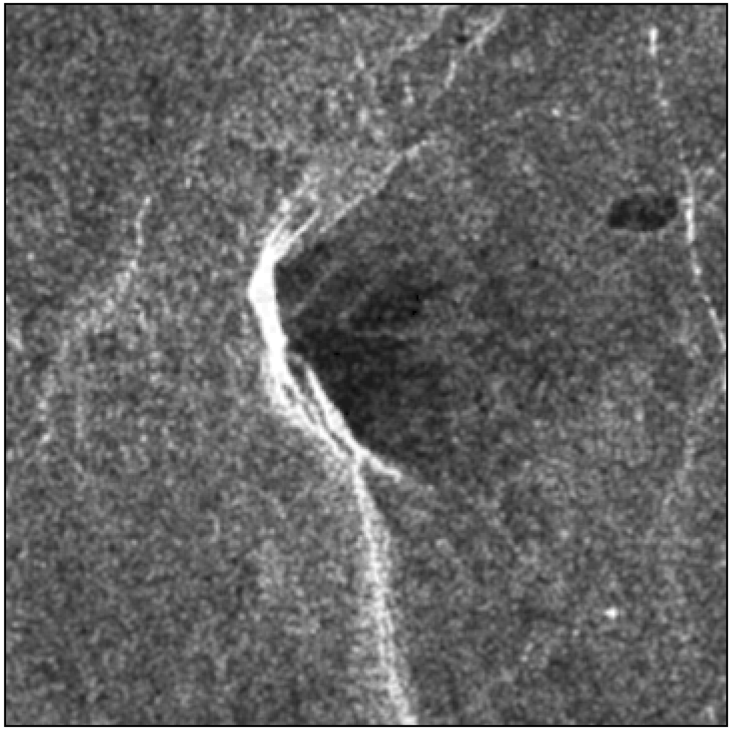
where

m,n	$=$ dimensions of filter kernel
I	$=$ mean pixel value
S	$=$ standard deviation of pixel values within filter window
T	$=$ absolute pixel distance from center pixel
$DAMP$	$=$ damping factor
M	$= \exp^{-A*T}$
A	$= DAMP * C_i^{\,2}$
C_i	$= S/I$

Speckle – Gamma MAP



Original



3x3 Gamma MAP

Speckle – Gamma MAP

- kernel math

$$\text{pixel value} = \begin{cases} I & \text{for } C_i \text{ less than or equal to } C_u \\ \dfrac{B * I + \sqrt{D}}{2 * \alpha} & \text{for } C_u < C_i < C_{max} \\ CP & \text{for } C_i \text{ greater than or equal to } C_{max} \end{cases}$$

where

N_{look} = number of looks

var = variance in filter window

CP = center pixel value

I = mean pixel value in the filter window

$C_u = 1 / \sqrt{N_{look}}$

$C_i = \sqrt{\text{var}} / I$

$C_{max} = \sqrt{2} * C_u$

$\alpha = \left(1 + c_u{}^2\right) / \left(c_i{}^2 - c_u{}^2\right)$

$B = \alpha - N_{look} - 1$

$D = I_2 * B_2 + 4 * \alpha * N_{look} * I * CP$

Applications

- detection of linear features and edges
  - faults
  - lineaments
  - roads
  - railways
  - runways etc.
- noise reduction
  - primarily SAR imagery

Conclusions

- there is no fit all applications filter
- your needs have to be very well defined
  - what is the filter supposed to do ?
  - there probably is a filter out there to meet your needs – finding it is sometimes a challenge
  - the solution to your problem might be a combination of different filters
- filtering always involves a <u>loss</u> of information

Questions

?