SPATIAL DOMAIN FILTERING
NONLINEAR FILTERS
Nonlinear Filtering

• As well as any linear filter, a nonlinear filter is defined by an operator
  \[ \hat{f}(x, y) = T(g(x, y)) \]
  where \( g(x, y) \) is a signal to be processed, \( T \) is a filtering operator and \( \hat{f}(x, y) \) is a resulting signal.

• Linearity does not hold for a nonlinear filter
  \[ T(a f(x, y) + b \eta(x, y)) \neq a T(f(x, y)) + b T(\eta(x, y)) \]
Nonlinear spatial filtering of an image of size $M \times N$ with a filter of size $m \times n$ is defined by the expression

$$\hat{f}(x, y) = T\left( S_{xy} \right)$$

where $S_{xy}$ is a local $m \times n$ window around the pixel $g(x, y)$ in the image $g$ to be processed, and $T$ is the nonlinear filtering operator.
Variational Series –
Series of Order Statistics

• An arrangement of the values of a random sample \( x_1, \ldots, x_n \) with distribution function \( F(x) \) in ascending sequence \( x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(n)} \) where

\[
x_{(1)} = \min_{i=1,\ldots,n} x_i ; \quad x_{(n)} = \max_{i=1,\ldots,n} x_i
\]

• The series is used to construct the empirical distribution function \( F_n(x) = m_x / n \), where \( m_x \) is the number of terms of the series which are smaller than \( x \).
Variational Series – Series of Order Statistics

- A rank (order statistics) of an element in a variational series is its serial number in the series

\[ x_{(1)} \leq x_{(2)} \leq \ldots \leq x_{(n)} \]

Rank 1  Rank 2  ...  Rank n
Variational Series – Series of Order Statistics

- **Example 1.** Let us have the following sample: 200, 101, 102, 125, 5, 10, 207, 180, 100
  Its variational series is 5, 10, 100, 101, 102, 180, 200, 201, 207

- **Example 2.** Let us have the following sample: 200, 101, 102, 101, 207, 101, 100, 207, 5
  Its variational series is 5, 100, 101, 101, 101, 102, 200, 207, 207
Median Filter – the Simplest Spatial Domain Nonlinear Filter

- Median filter replaces an intensity value in each pixel by a local median taken over a local $n \times m$ processing window:

$$\hat{f}(x, y) = \text{MED}\left(S_{xy}\right)$$

where $S_{xy}$ is a local $m \times n$ window around the pixel $(x,y)$ in the image $g$ (to be processed), and MED is the median value taken over the $S_{xy}$ window.
Median Filter – Implementation

• To implement the median filter with an \( n \times m \) processing window, it is necessary to build a variational series from the elements of the processing window \( S_{xy} \) around the pixel \((x,y)\) (to be processed)

• The central element of the variational series is the median of the intensity values in the \( S_{xy} \) window
Median Filter and Impulse Noise

• Median filter is highly efficient for impulse noise filtering
• Median filter with 3x3 window can almost completely remove impulse noise with the corruption rate up to 30%
• Applied iteratively, it can remove even noise with a higher corruption rate
Median Filter and Impulse Noise

• The ability of the median filter to remove impulse noise is based on the following:
• Impulses have unusually high (or low) intensity values compared to their neighbors
• This means that they are located in the left or right ends of the variational series built from the intensity values in a local window $S_{xy}$ around the pixel (x,y)
Median Filter and Impulse Noise

3x3 window, 201 is the impulse

Variational Series

The processing result
The impulse 201 was removed
Mean Filter and Impulse Noise

• Unlike the median filter, the mean filter is unable to remove impulse noise. It only “washes” impulses.

3x3 window, 201 is the impulse

<table>
<thead>
<tr>
<th>25</th>
<th>36</th>
<th>36</th>
</tr>
</thead>
<tbody>
<tr>
<td>47</td>
<td>201</td>
<td>48</td>
</tr>
<tr>
<td>104</td>
<td>110</td>
<td>120</td>
</tr>
</tbody>
</table>

Mean = \[ \frac{25 + 36 + 36 + 47 + 201 + 48 + 104 + 110 + 120}{9} = 81 \]

The processing result
The impulse 201 was just “washed”
Disadvantages of Median Filtering

• Median filter removes impulse noise, but it also smooths ("washes") all edges and boundaries and may "erase" all details whose size is about $\frac{n}{2} \times \frac{m}{2}$, where $n \times m$ is a window size.

• As a result, an image becomes "fuzzy."

• Median filter is not so efficient for additive Gaussian noise removal, it yields to linear filters.
Detection of Impulse Noise

• To reduce a image smoothing by the median filter, impulse detectors should be used

• A detector analyses local statistical (and possibly other) characteristics in a local window around each pixel using some criteria, and marks those pixels that are corrupted by noise

• Then only marked pixels shall be processed by the median filter
Differential Rank Impulse Detector

• This detector is based on the following reasonable assumptions
  ➢ Impulses are located either in the ends of the variational series (salt-end-pepper and bipolar noise) or close to these ends (random noise)
  ➢ The difference between the intensity values of impulse and its neighbor located in the variational series between this impulse and the median should not exceed some reasonable threshold value
Differential Rank Impulse Detector

- Let $R(x)$ be the rank of an element $x$ in the variational series and $\text{var}(k)$ be the intensity value of the pixel whose rank is $k$

$$d_{xy} = \begin{cases} 
  g(x, y) - \text{var}\left[R(g(x, y)) - 1\right], & \text{if } R(g(x, y)) > R\left(\text{MED}_{S_{xy}}\right) \\
  g(x, y) - \text{var}\left[R(g(x, y)) + 1\right], & \text{if } R(g(x, y)) < R\left(\text{MED}_{S_{xy}}\right) \\
  0, & \text{otherwise,}
\end{cases}$$
Differential Rank Impulse Detector

• Let $r$ be the length (in ranks) of the interval in the variational series where an impulse can be located.

• Let $s$ be the maximal acceptable difference between the intensity value of the pixel of interest and its neighbor located in the variational series between this impulse and the median.

\[
 d_{xy} = \text{Var}(\text{Pixel}) - \text{Var}(\text{Neighbor}) \geq s
\]

Differential Criteria of an impulse
Differential Rank Impulse Detector

• Differential Rank Impulse Detector (DRID) works as follows

• The pixel \((x, y)\) is considered noisy if the following condition holds:

\[
\left( \left( R(g(x, y) \leq r) \lor \left( R(g(x, y)) \geq mn - r + 1 \right) \right) \land \left( d_{xy} \geq s \right) \right)
\]

the intensity value \(g(x, y)\) in the pixel of interest is located in the \(r\)-interval from one of the ends of the variational series, and the difference \(d_{xy}\) between this value and its neighbor located in the variational series closer to the median exceeds \(s\)
Filtering of Scratches

• Any scratch is a collection of impulses – there are consecutive impulses extended in some direction along some virtual curve
• To remove a scratch from an image, it is necessary to process it locally and to use the mask median filter
Filtering of Scratches

• In the mask median filter, a median is taken not over all the pixels in a filtering window, but only over those marked in the mask.

• To remove a scratch, it is necessary to create a “counterscratch” mask including there pixels from the virtual perpendicular to the scratch.
Filtering of Scratches

Scratch

Mask
Threshold Boolean Filtering (TBF)

- Threshold Boolean filtering (also often referred to as stack filtering) is reduced to:
  - Splitting an image with M+1 gray levels into M binary planes (slices) by thresholding
  - Separate processing of the binary planes (slices) using a processing Boolean function
  - Merging the processed binary planes into a resulting image
Threshold Boolean Filtering (TBF)

- Let $g$ be a scale image with $(M+1)$ gray levels

- Let $g^{(k)}(x, y) = \begin{cases} 1 & \text{if } f(x, y) \geq k \\ 0 & \text{otherwise} \end{cases}$; $k = 1, \ldots, M$ be a $k^{th}$ binary plane

- Then a filtering operation is determined by

$$\hat{f}(x, j) = \sum_{k=1}^{M} F \left( S^{(k)}_{xy} \right)$$

where $F$ is the processing Boolean function, $S^{(k)}_{ij}$ is a $k^{th}$ binary plane of a local window around a pixel of interest
Threshold Boolean Filter (TBF)

The idea behind the Threshold Boolean Filter is to process each binary plane of an image separately, and after filtering all the binary planes are combined together to form the resulting image. The following diagram illustrates this idea:

\[ g^{(k)}(x, y) = \begin{cases} 1, & \text{if } g(x, y) \geq k \\ 0, & \text{otherwise} \end{cases} \]

\[ k = 1, 2, ..., M \] – the binary slice index
Impulse Noise Filtering using TBF

The filter is defined by the following functions. For 3x3 window:

\[
f(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9) = \left[ x_5 \wedge \left( \bigwedge_{j=1}^{T} \left( \bigvee_{k=i_{j_1}}^{i_{j_t}} x_k \right) \right) \right] \lor \left[ \overline{x_5} \lor \left( \bigvee_{j=1}^{T} \left( \bigwedge_{k=i_{j_1}}^{i_{j_k}} x_k \right) \right) \right]
\]

For 5x5 window

\[
f_{5x5}(x_1, \ldots, x_{25}) = f(\ldots) \wedge \left[ x_5 \wedge \left( \bigwedge_{j=1}^{P} \left( \bigvee_{k=i_{j_1}}^{i_{j_s}} x_k \right) \right) \right] \lor \left[ \overline{x_5} \lor \left( \bigvee_{j=1}^{P} \left( \bigwedge_{k=i_{j_1}}^{i_{j_k}} x_k \right) \right) \right]
\]

where \(X_5\) is the central element of the window, i.e. the pixel under consideration,
\(T\) is the number of all possible elementary conjunctions of \(t\) variables out of 8 (the number of pixels in a 3x3 window),
\(P\) is the number of all possible elementary conjunctions of \(s\) variables out of 16 (5x5 window).
Impulse Noise Filtering using TBF

• TBF is highly efficient for filtering of random impulse noise filtering with a low corruption rate (≤5%). It may preserve image details with a higher accuracy than other filters

• Disadvantages: strictly dependent on the successful choice of the parameters and computationally costly (compared to median and rank-order ER filters, even used in conjunction with noise detector)
Rank-Order (Order Statistic) Filters

• Rank-order filtering is based on the analysis of the variational series followed by some nonlinear averaging of a signal
• The median filter is a representative of the rank-order filters family
• We will consider rank-order filters using their classification done by Leonid Yaroslavsky
Rank-Order EV Filter

- Let $\epsilon_v$ be a value, which determines the following interval ($EV$) around the intensity value of the pixel of interest $g(x, y)$

$$EV = \left\{ g(i, j) : |g(x, y) - g(i, j)| \leq \epsilon_v ; g(i, j) \in S_{xy} \right\}$$

- An EV interval contains those intensity values whose difference from the intensity value in the pixel of interest does not exceed $\epsilon_v$

- A suboptimal value of the parameter is $\epsilon_v = \sigma$ - standard deviation measured over an image
Rank-Order EV Filter

• Rank-order EV filter is determined as follows

\[
\hat{f}(x, y) = \begin{cases} 
\text{MEAN}(g(i, j)), g(i, j) \in EV \\
\text{or} \\
\text{MED}(g(i, j)), g(i, j) \in EV 
\end{cases}
\]

• The first option leads to more careful and accurate reduction of additive noise (compared to linear filters)
Rank-Order EV Filter

• Rank-order EV filter preserves small details and image boundaries with a higher accuracy than linear filters. With a careful choice of $E_v$, it is possible to avoid smoothing of $n/2 \times m/2$ details.

• Disadvantage is the dependence on the choice of the parameter $E_v$.

• This filter may show better results if EV has been chosen adaptively for each $S_{xy}$ window.
• Let $\varepsilon_r$ is a value, which determines the following interval (ER) around the intensity value of the pixel of interest

$$ER = \left\{ g(i, j) : \left| R(g(x, y)) - R(g(i, j)) \right| \leq \varepsilon_r; g(i, j) \in S_{xy} \right\}$$

• An ER interval contains those intensity values whose ranks differ from the rank of the intensity value in the pixel of interest by not more than $\varepsilon_r$. 
Rank-Order ER Filter

• Rank-order ER filter is determined as follows

\[
\hat{f}(x, y) = \begin{cases} 
\text{MEAN}(g(i, j)), g(i, j) \in ER \\
\text{or} \\
\text{MED}(g(i, j)), g(i, j) \in ER
\end{cases}
\]

• The second option leads to more careful removal of random impulse noise with a low corruption rate (1-5%) (the filter should be applied iteratively). It can also be used to “close” impulse gaps (spots) – a window larger than a spot should be used

• The second option is also better for speckle (multiplicative) noise
Rank-Order ER Filter

• Advantage of the rank-order ER filter is that it preserves small details and image boundaries with a higher accuracy than the median filter. With a careful choice of $\mathcal{E}_r$, it is possible to remove salt-and-pepper impulse noise iteratively with more careful preservation of image details.

• Disadvantage is the dependence on the choice of the parameter ER.

• This filter may show better results if ER has been chosen adaptively for each $S_{xy}$ window.
Rank-Order KNV Filter

• Let $k$ be a value, which determines the following interval (KNV) around the intensity value of the pixel of interest

$$
\text{KNV} = \left\{ g_s = g(i, j) : \left| R(g(x, y)) - R(g(i, j)) \right| \leq nm; \right\}
$$

• A KNV interval contains $k$ closest pixels from $S_{xy}$ to the $xy^{th}$ pixel, in terms of the intensity values closest to the intensity $g(x, y)$
Rank-Order KNV Filter

• Rank-order KNV filter is determined as follows

\[
\hat{f}(x, y) = \begin{cases} 
\text{MEAN}(g(i, j)), g(i, j) \in KNV \\
\text{or} \\
\text{MED}(g(i, j)), g(i, j) \in KNV 
\end{cases}
\]

• The first option leads to more careful reduction of additive noise (compared to linear filters)
Rank-Order KNV Filter

• Advantage of the rank-order KNV filter is that it preserves small details whose area is $< k$ squared pixels

• However, its ability to reduce noise is limited. It is better to use this filter only when it is absolutely necessary to preserve some small details

• Another disadvantage is its dependence on the choice of the parameter $k$

• This filter may show better results if $k$ has been chosen adaptively for each $S_{xy}$ window
Geometric Mean Filter

\[ \hat{f}(x, y) = \left( \prod_{(s,t) \in S_{xy}} S(s,t) \right)^{\frac{1}{mn}} \]

- Can preserve details a little bit better than the arithmetic mean filter
Harmonic Mean Filter

\[ \hat{f}(x, y) = \frac{mn}{\sum_{(s,t) \in S_{xy}} \frac{1}{g(s, t)}} \]

- Can be good for multiplicative (speckle) noise removal
Contraharmonic Mean Filter

\[ \hat{f}(x, y) = \frac{\sum_{(s,t) \in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t) \in S_{xy}} g(s,t)^Q} \]

where \( Q \) is the order of the filter

- With positive \( Q \) eliminates pepper noise, while with negative \( Q \) eliminates salt noise
- With \( Q=0 \) reduces to the arithmetic mean filter, and with \( Q=1 \) – to the harmonic mean filter