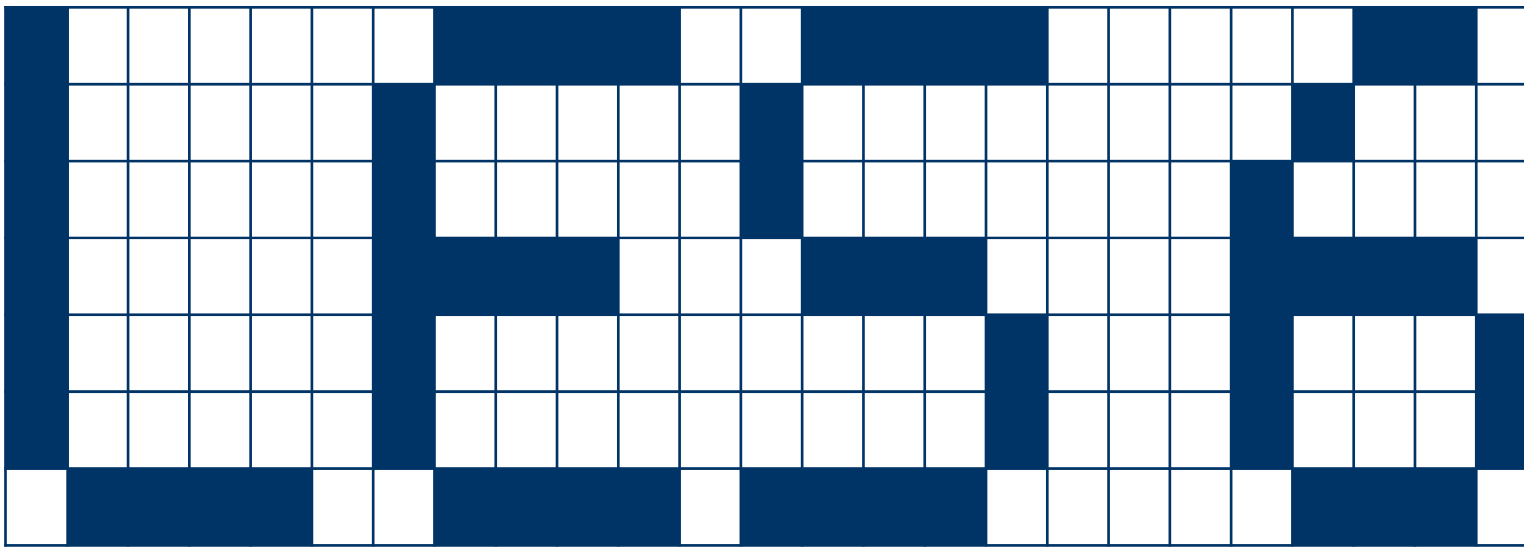
EVD1 – Vision operators



Nonlinear filters

Nonlinear filters:
- Operate on greyscale images
- Are mainly used to remove noise
- But there are also other uses (dilation, erosion, edge detection)

Properties of Nonlinear filters:
- Computes a nonlinear function over a window of pixels
- Replaces one pixel within this window
- Window size is always odd, so there is a center pixel

Examples of Nonlinear filters:
- Average filter
- Harmonic filter
- Maximum filter
- Median filter
- Midpoint filter
- Minimum filter
- Range filter

Nonlinear filters

Function prototype:

```
typedef enum
{
  AVERAGE = 0,
  HARMONIC,
  MAX,
  MEDIAN,
  MIDPOINT,
  MIN,
  RANGE
}eFilterOperation;


void vNonlinearFilter(image_t *src,
                      image_t *dst,
                      eFilterOperation fo,
                      uint8_t n) // n equals the mask size
                                 // n should be an odd number
```

Nonlinear filters: Average

Average filter: Calculates the arithmetic mean of the pixels within the window

Window size (n) = 3

	0					
0	1	1	3	1	…	…
	2	3	3	2	…	…
	2	1	3	1	…	…
	0	2	2	1	…	…

$$p_{(y,x)} = \frac{1}{n^2} \sum_{i=-n/2}^{i=n/2} \sum_{j=-n/2}^{j=n/2} p(y+i, x+j)$$

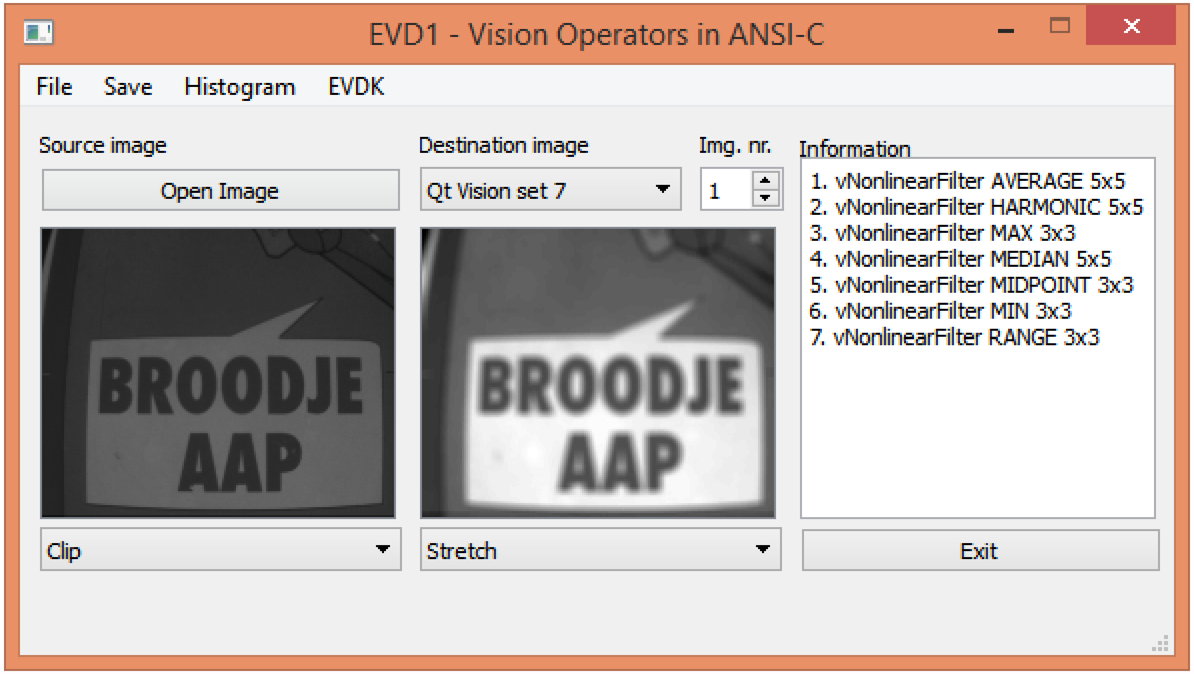
$$p_{(1,1)} = \frac{1}{3^2} \times (1+1+3+2+3+9+2+1+3) = 2{,}7$$

$$p_{(1,2)} = \frac{1}{3^2} \times (1+3+1+3+9+2+1+3+1) = 2{,}7$$

Nonlinear filters: Average

Average filter:

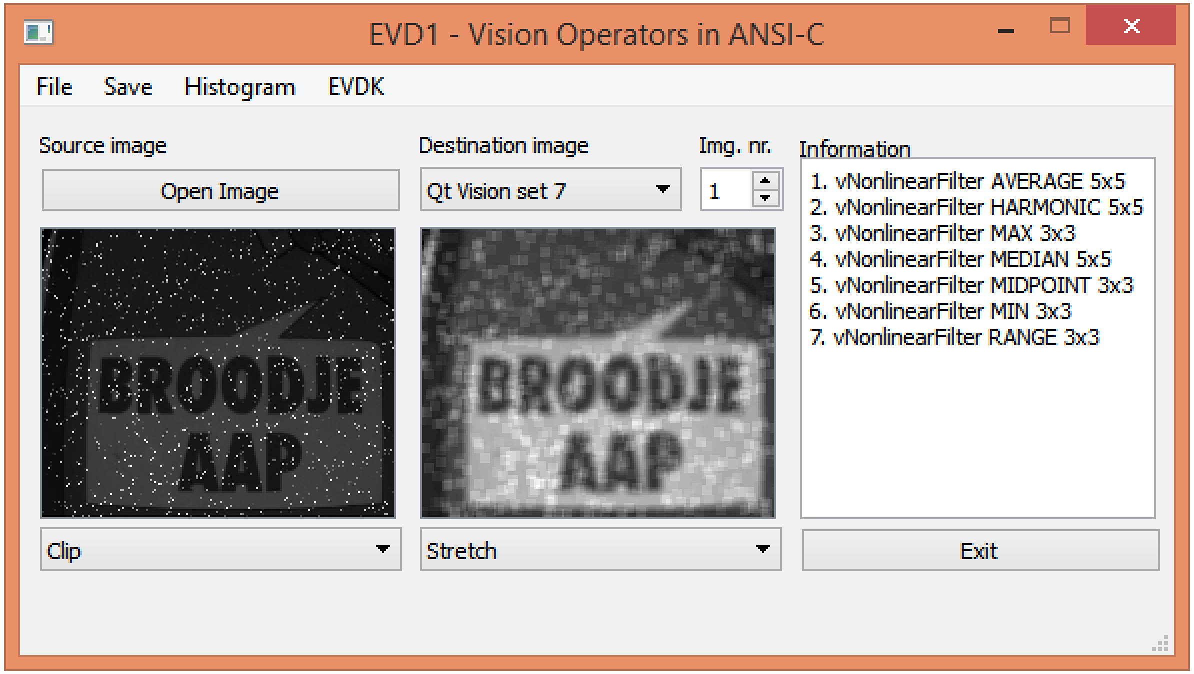
- Used to remove single noise pixels
- High noise values infuence all neighbour pixels
- Larger window means more blurring effect and loss of edge information



Nonlinear filters: Average

Average filter:

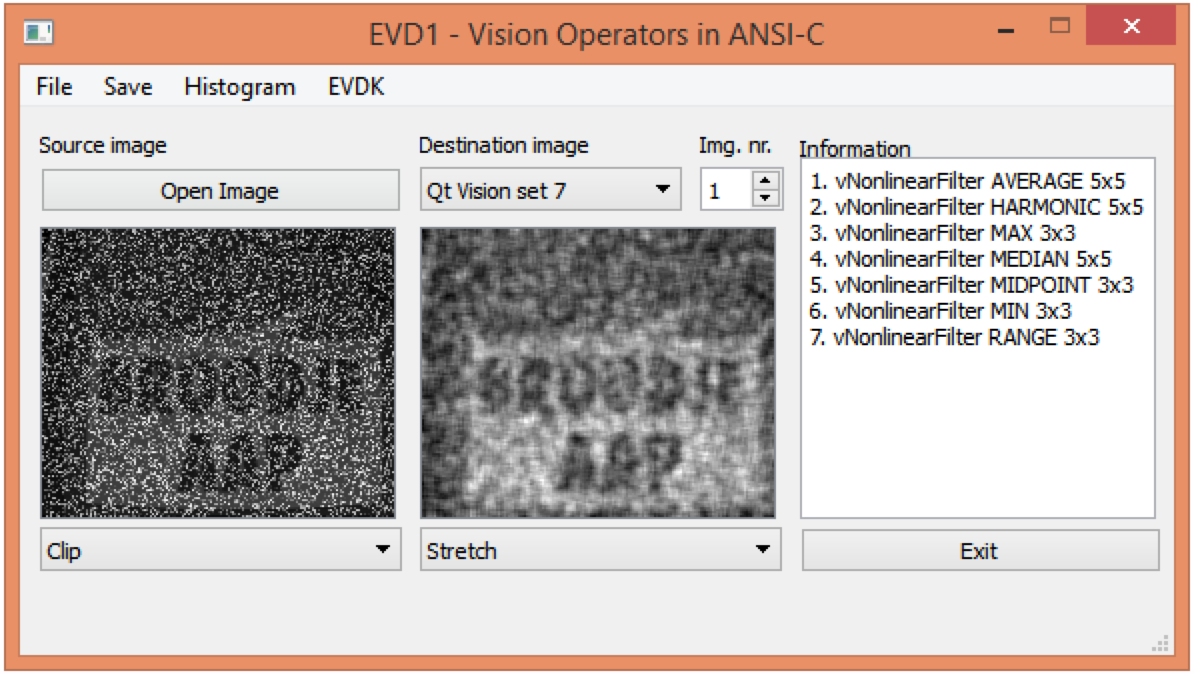
- **Used to remove single noise pixels**
- **High noise values infuence all neighbour pixels**
- **Larger window means more blurring effect and loss of edge information**



Nonlinear filters: Average

Average filter:

- **Used to remove single noise pixels**
- **High noise values infuence all neighbour pixels**
- **Larger window means more blurring effect and loss of edge information**



Nonlinear filters: Harmonic

Harmonic mean filter: Calculates the average of rates of the pixels within the window

	0					
0	1	1	3	1	…	…
	2	2	2	2	…	…
	2	1	3	1	…	…
	0	2	2	1	…	…

$$p_{(y,x)} = \frac{n^2}{\sum_{i=-n/2}^{i=n/2} \sum_{j=-n/2}^{j=n/2} \left(\frac{1}{p(y+i, x+j)}\right)}$$

er gets

What to do?!

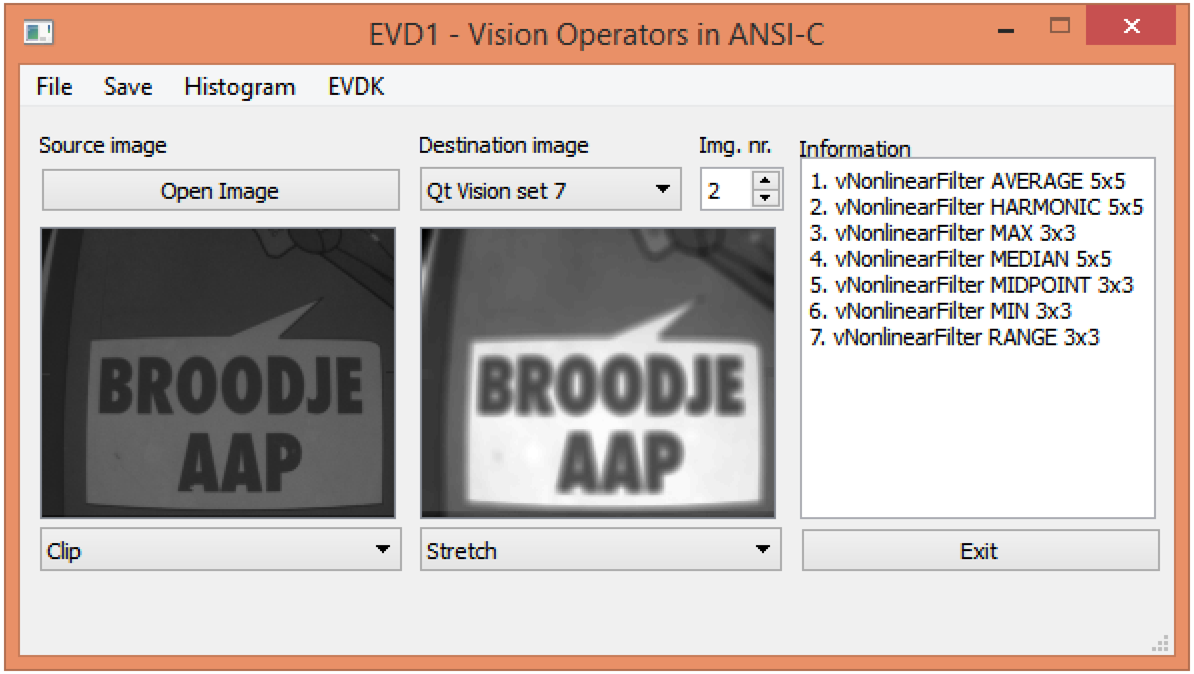
$$p_{(1,1)} = \frac{3^2}{\frac{1}{1}+\frac{1}{1}+\frac{1}{3}+\frac{1}{2}+\frac{1}{3}+\frac{1}{9}+\frac{1}{2}+\frac{1}{1}+\frac{1}{3}} = 1{,}8$$

$$p_{(1,2)} = \frac{3^2}{\frac{1}{1}+\frac{1}{3}+\frac{1}{1}+\frac{1}{3}+\frac{1}{9}+\frac{1}{2}+\frac{1}{1}+\frac{1}{3}+\frac{1}{1}} = 1{,}6$$

Nonlinear filters: Harmonic

Harmonic mean filter:

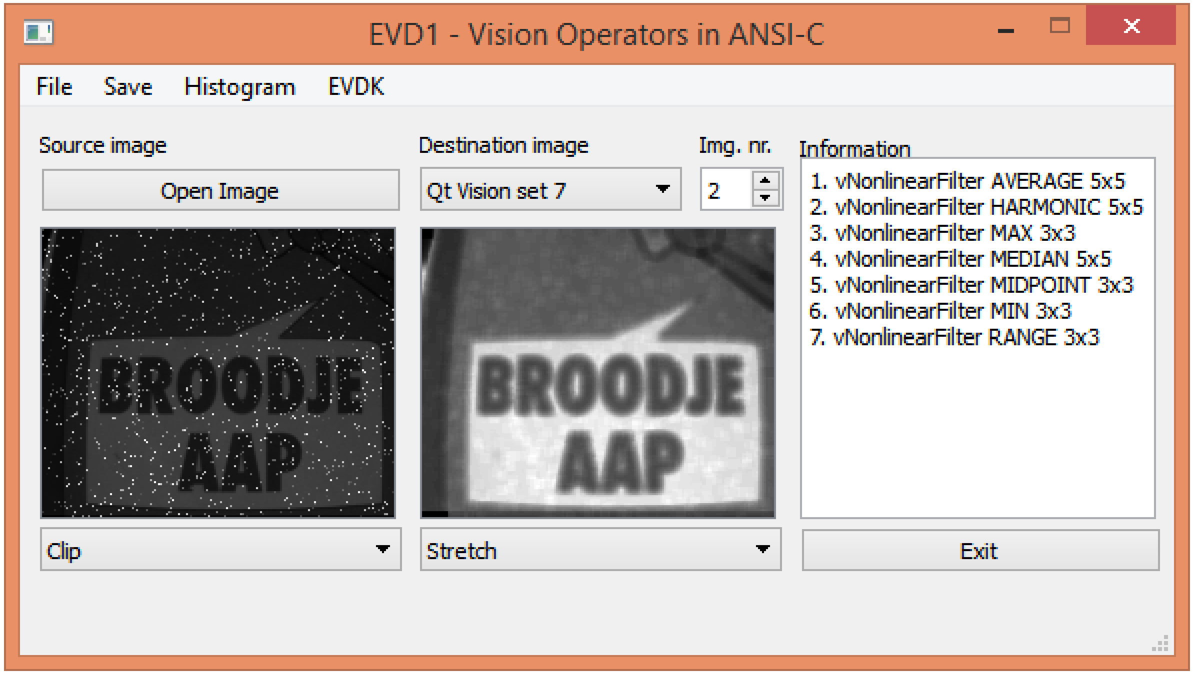
- **Used to remove positive outlier noise pixels**
- **Larger window means more blurring effect and loss of edge information**



Nonlinear filters: Harmonic

Harmonic mean filter:

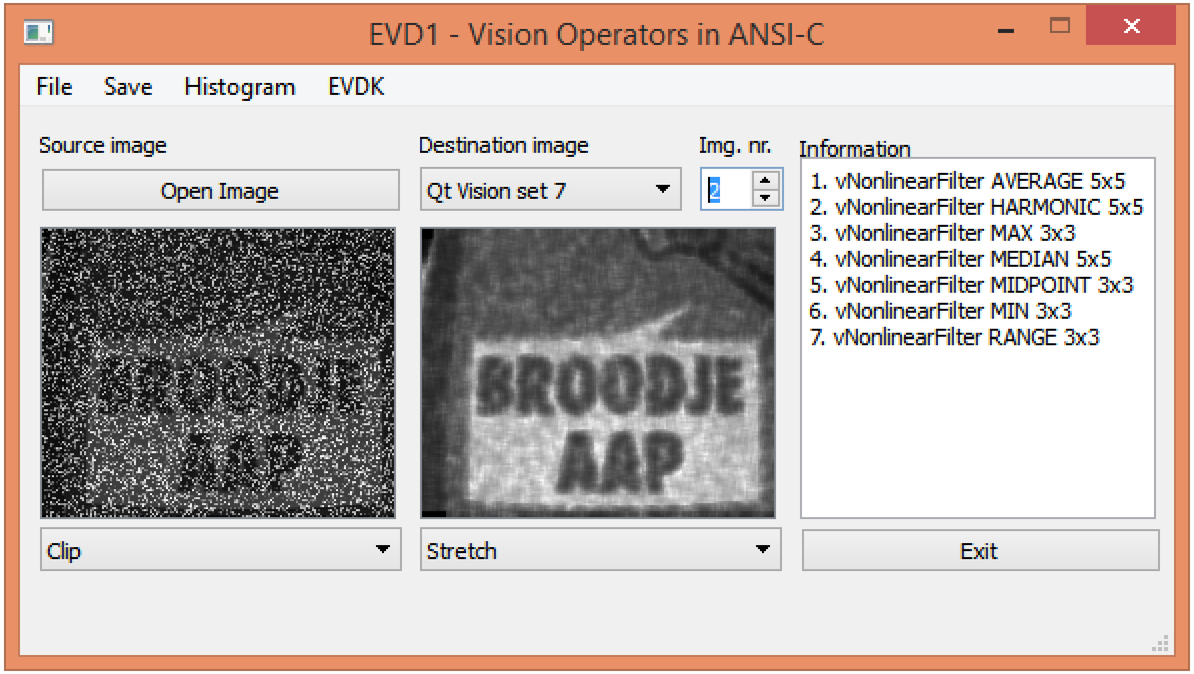
- **Used to remove positive outlier noise pixels**
- **Larger window means more blurring effect and loss of edge information**



Nonlinear filters: Harmonic

Harmonic mean filter:

- **Used to remove positive outlier noise pixels**
- **Larger window means more blurring effect and loss of edge information**



Nonlinear filters: Max

Max filter: Selects the maximum pixel value from the window

	0					
0	1	1	3	1	…	…
	2	9	9	2	…	…
	2	1	3	1	…	…
	0	2	2	1	…	…

$$p_{(y,x)} = max\ [window(y+i, x+j)]_{i=-n/2\ j=-n/2}^{i=n/2\ j=n/2}$$

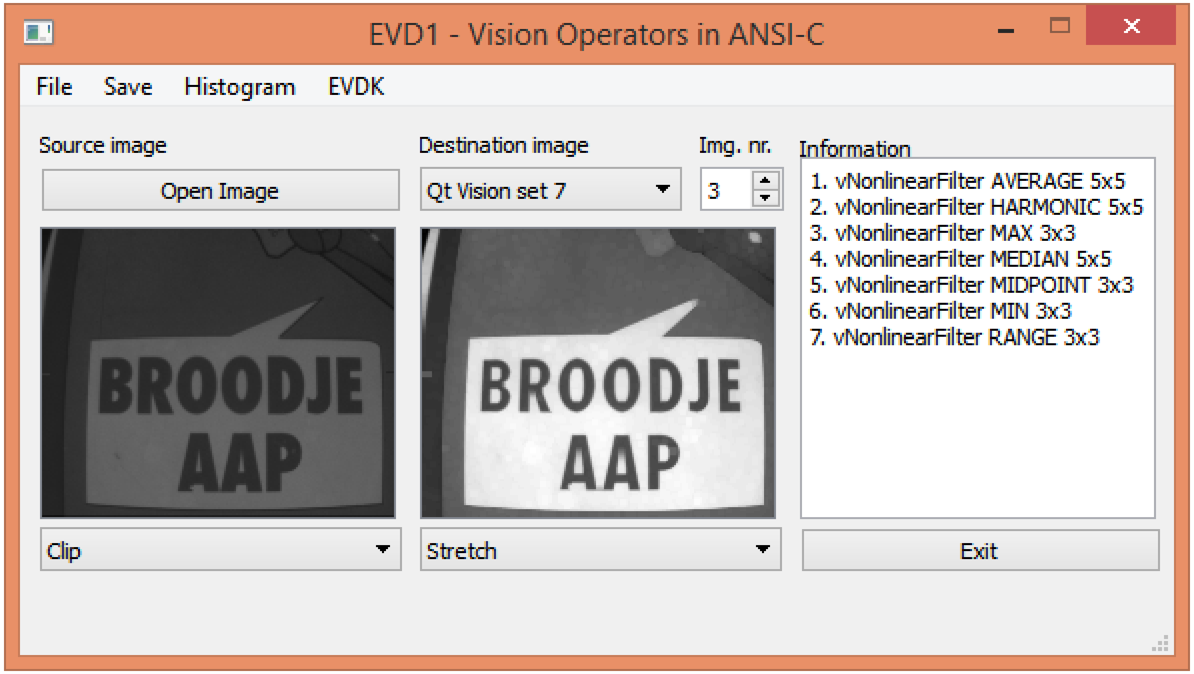
$$p_{(1,1)} = \max\ [1,1,3,2,3,9,2,1,3] = 9$$

$$p_{(1,2)} = \max\ [1,3,1,3,9,2,1,3,1] = 9$$

Nonlinear filters: Max

Maximum filter:

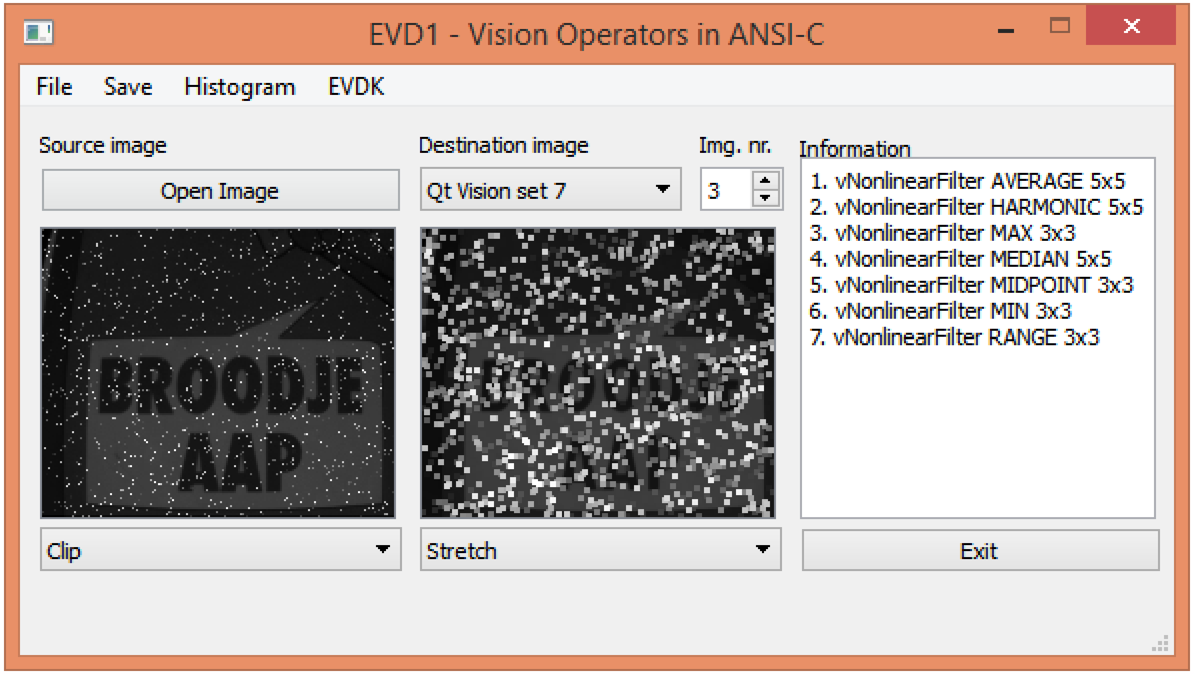
- **Used to remove <u>lower</u> value noise pixels**
- **Can also be used for binary dilation**



Nonlinear filters: Max

Maximum filter:

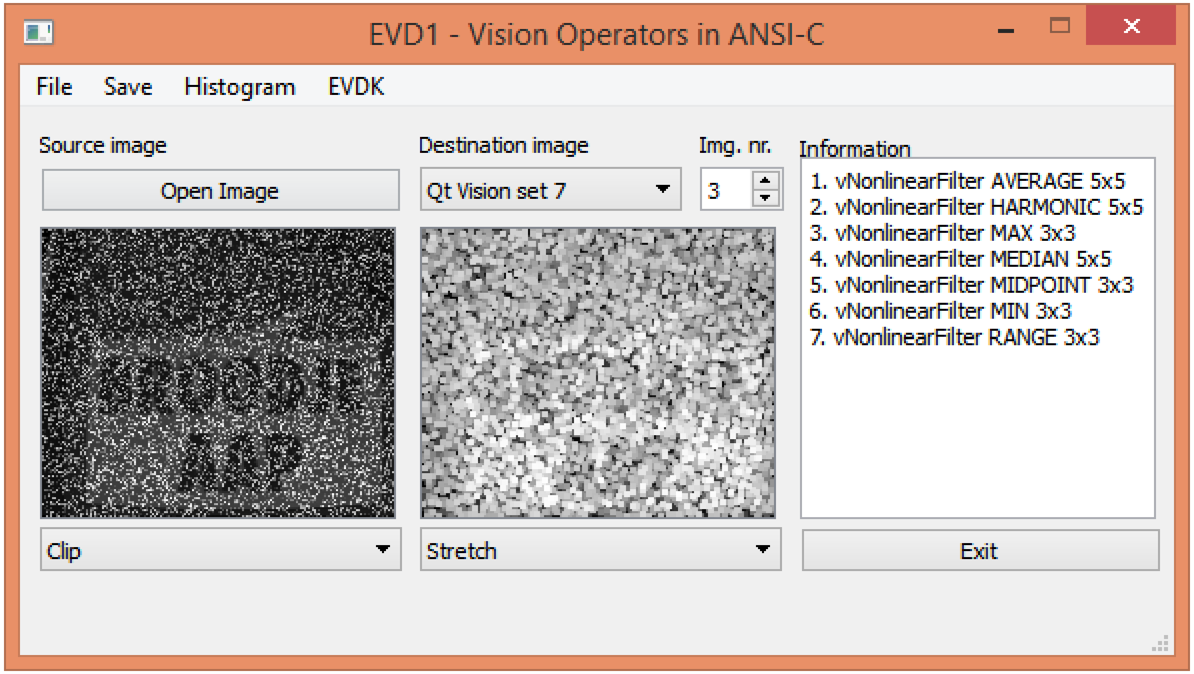
- **Used to remove <u>lower</u> value noise pixels**
- **Can also be used for binary dilation**



Nonlinear filters: Max

Maximum filter:

- **Used to remove <u>lower</u> value noise pixels**
- **Can also be used for binary dilation**



Nonlinear filters: Min

Min filter: Selects the minimum pixel value from the window

	0					
0	1	1	3	1	…	…
	2	1	1	2	…	…
	2	1	3	1	…	…
	0	2	2	1	…	…

$$p_{(y,x)} = min\left[window(y+i, x+j)\right]_{i=-^{n}/_{2}\ j=-^{n}/_{2}}^{i=^{n}/_{2}\ j=^{n}/_{2}}$$

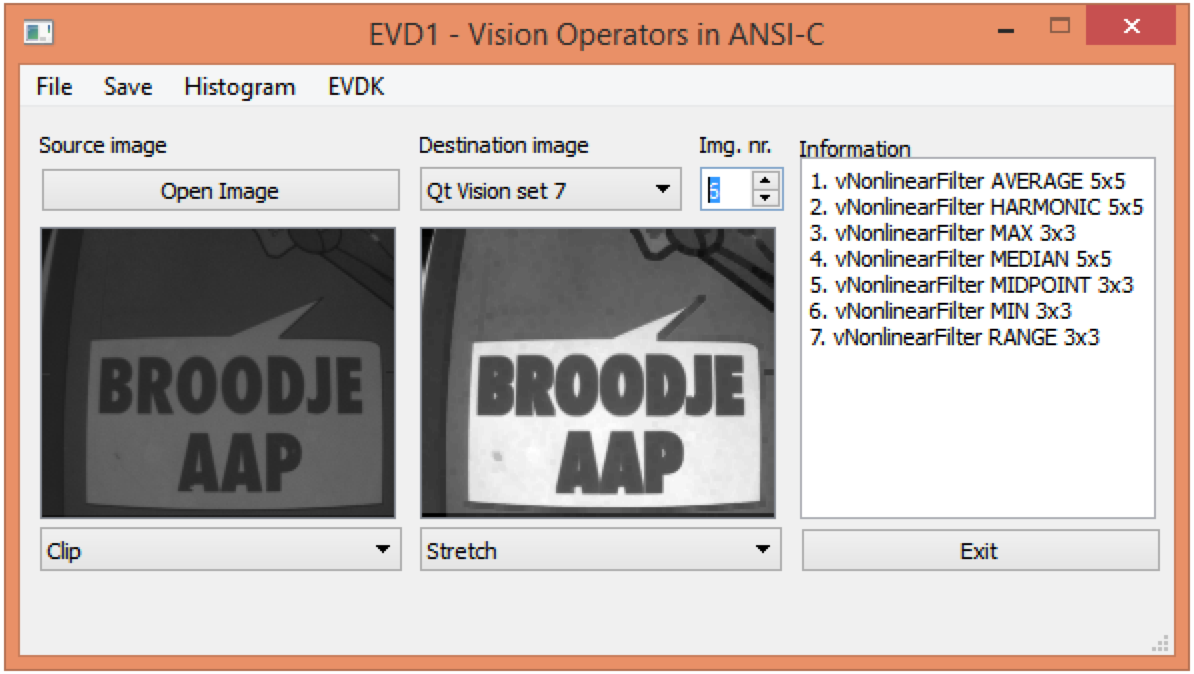
$$p_{(1,1)} = \min\left[1,1,3,2,3,9,2,1,3\right] = 1$$

$$p_{(1,2)} = \min\left[1,3,1,3,9,2,1,3,1\right] = 1$$

Nonlinear filters: Min

Minimum filter:

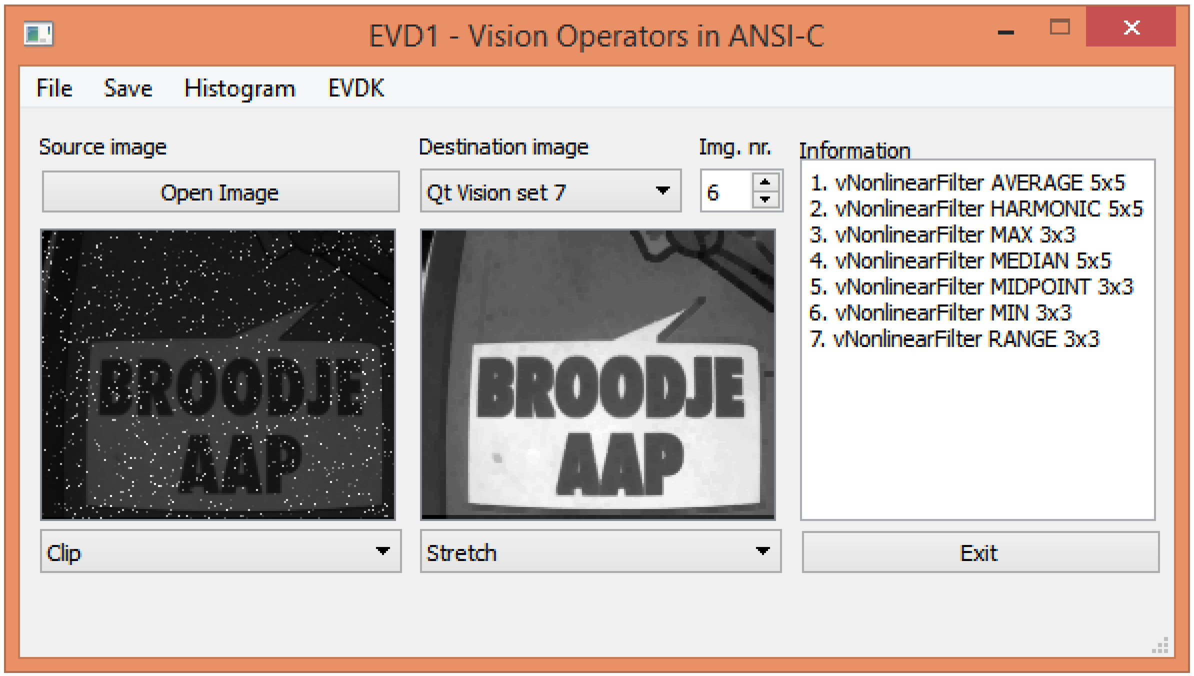
- **Used to remove <u>positive</u> value noise pixels**
- **Can also be used for binary erosion**



Nonlinear filters: Min

Minimum filter:

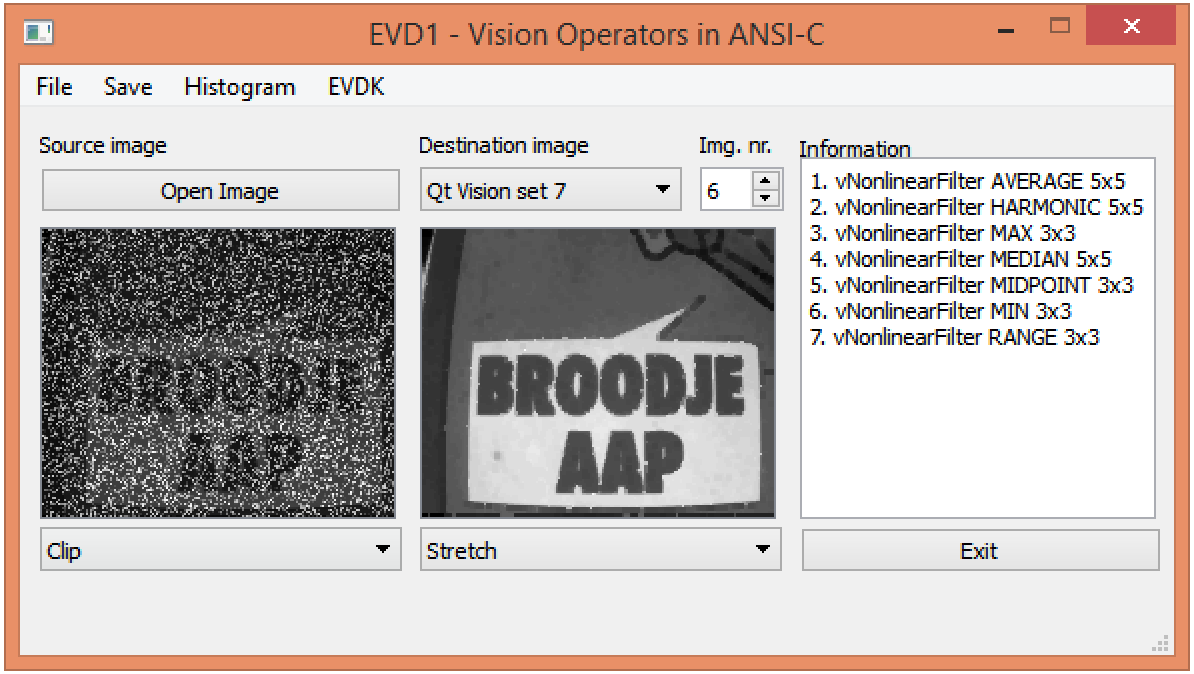
- **Used to remove <u>positive</u> value noise pixels**
- **Can also be used for binary erosion**



Nonlinear filters: Min

Minimum filter:

- Used to remove positive value noise pixels
- Can also be used for binary erosion



Nonlinear filters: Midpoint

Midpoint filter: Calculates the average of the maximum and minimum pixel value within the window

	0					
0	1	1	3	1	…	…
	2	5	5	2	…	…
	2	1	3	1	…	…
	0	2	2	1	…	…

$$p_{(1,1)} = \frac{9+1}{2} = 5$$

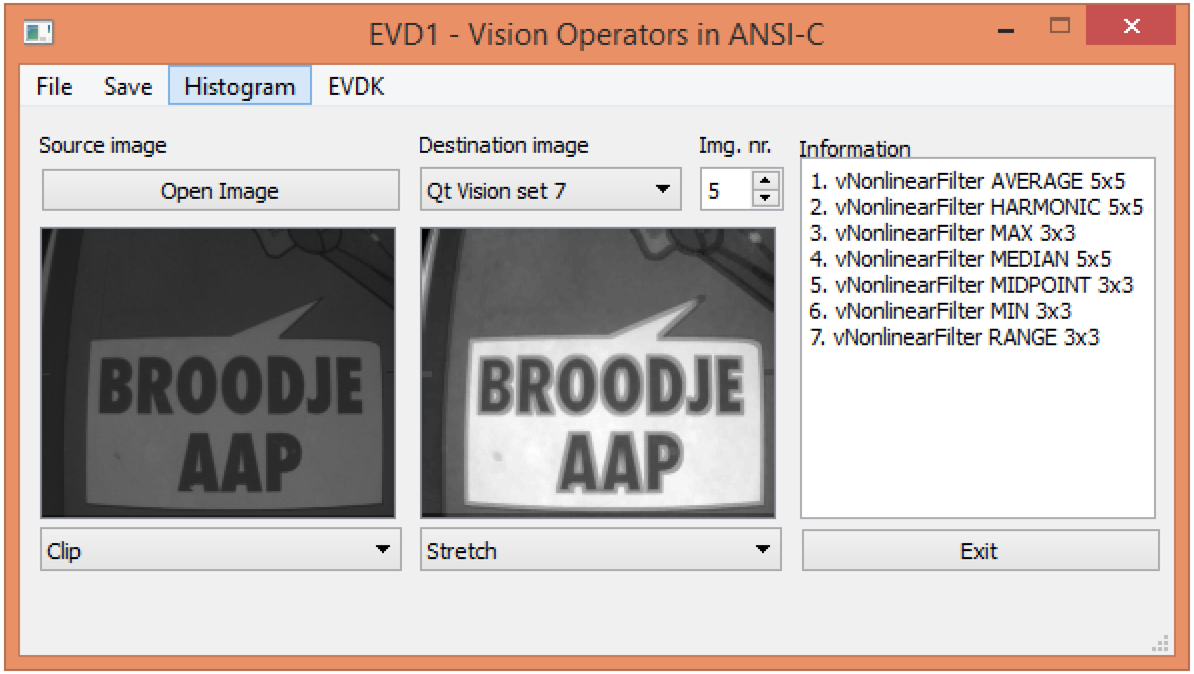
$$p_{(1,2)} = \frac{9+1}{2} = 5$$

$$p_{(y,x)} = \frac{max\,[window(y+i,x+j)]_{i=-^{n}/_{2}\;j=-^{n}/_{2}}^{i=^{n}/_{2}\;j=^{n}/_{2}} + min\,[window(y+i,x+j)]_{i=-^{n}/_{2}\;j=-^{n}/_{2}}^{i=^{n}/_{2}\;j=^{n}/_{2}}}{2}$$

Nonlinear filters: Midpoint

Midpoint filter:

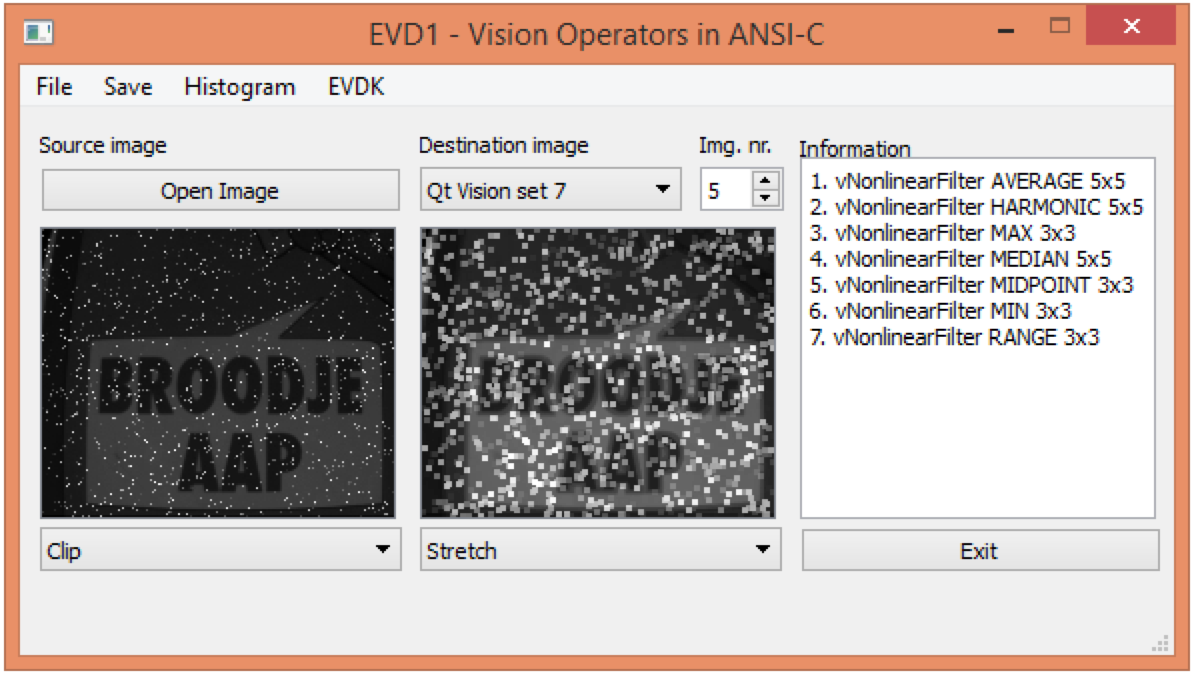
- **Used to remove short tailed noise pixels**



Nonlinear filters: Midpoint

Midpoint filter:

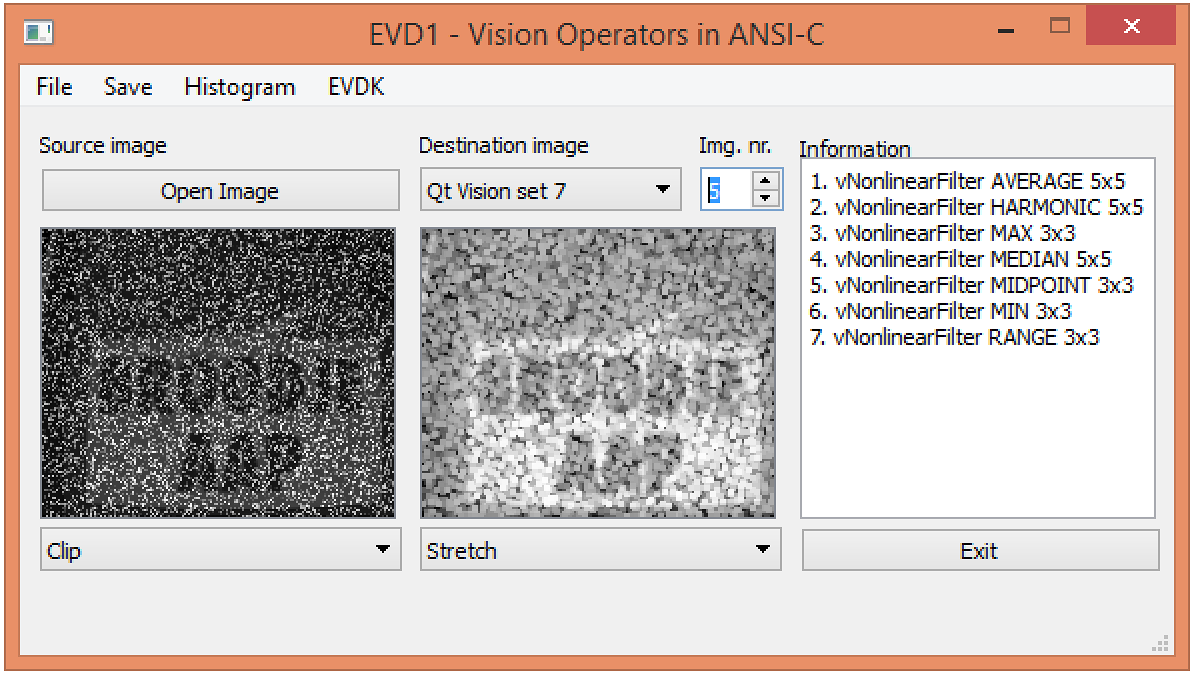
- **Used to remove short tailed noise pixels**



Nonlinear filters: Midpoint

Midpoint filter:

- **Used to remove short tailed noise pixels**



Nonlinear filters: Median

Median filter: Selects the median pixel value from the window
(by sorting the pixels and selecting the center value)

	0					
0	1	1	3	1	…	…
	2	2	2	2	…	…
	2	1	3	1	…	…
	0	2	2	1	…	…

$$p_{(y,x)} = median\left[window(y+i, x+j)\right]_{i=-n/2\ j=-n/2}^{i=n/2\ j=n/2}$$

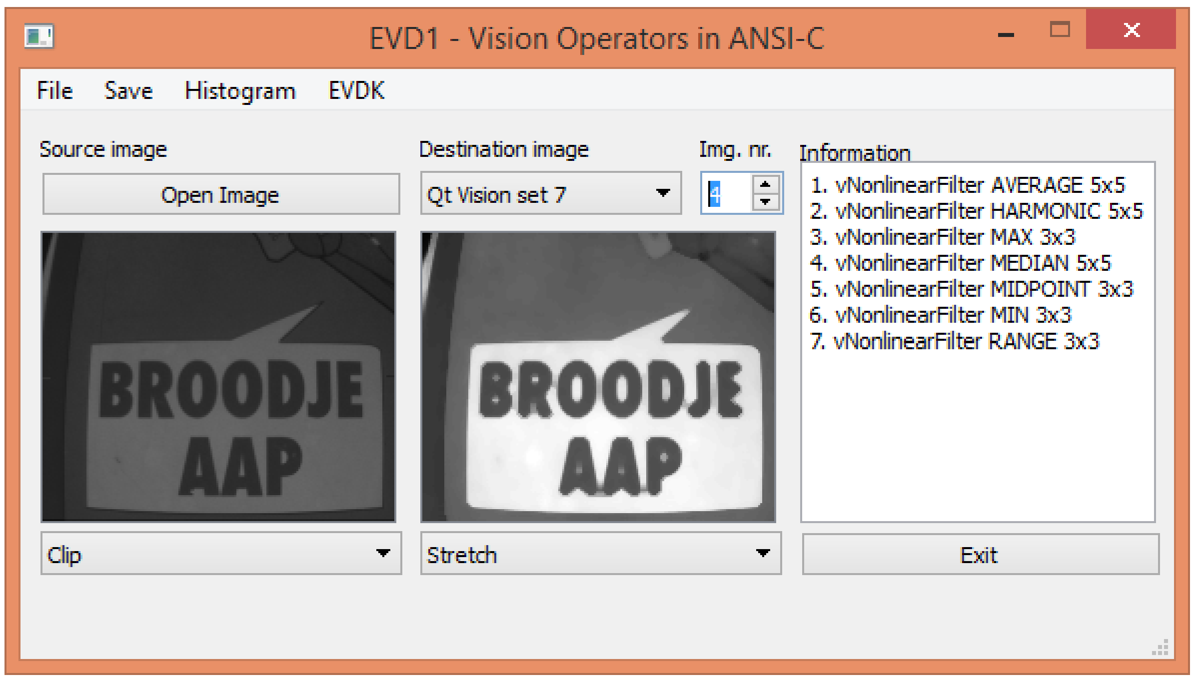
$$p_{(1,1)} = \text{median}[1,1,3,2,3,9,2,1,3] = median[1,1,1,2,2,3,3,3,9] = 2$$

$$p_{(1,2)} = \text{median}[1,3,1,3,9,2,1,3,1] = median[1,1,1,1,2,3,3,3,9] = 2$$

Nonlinear filters: Median

Median filter:

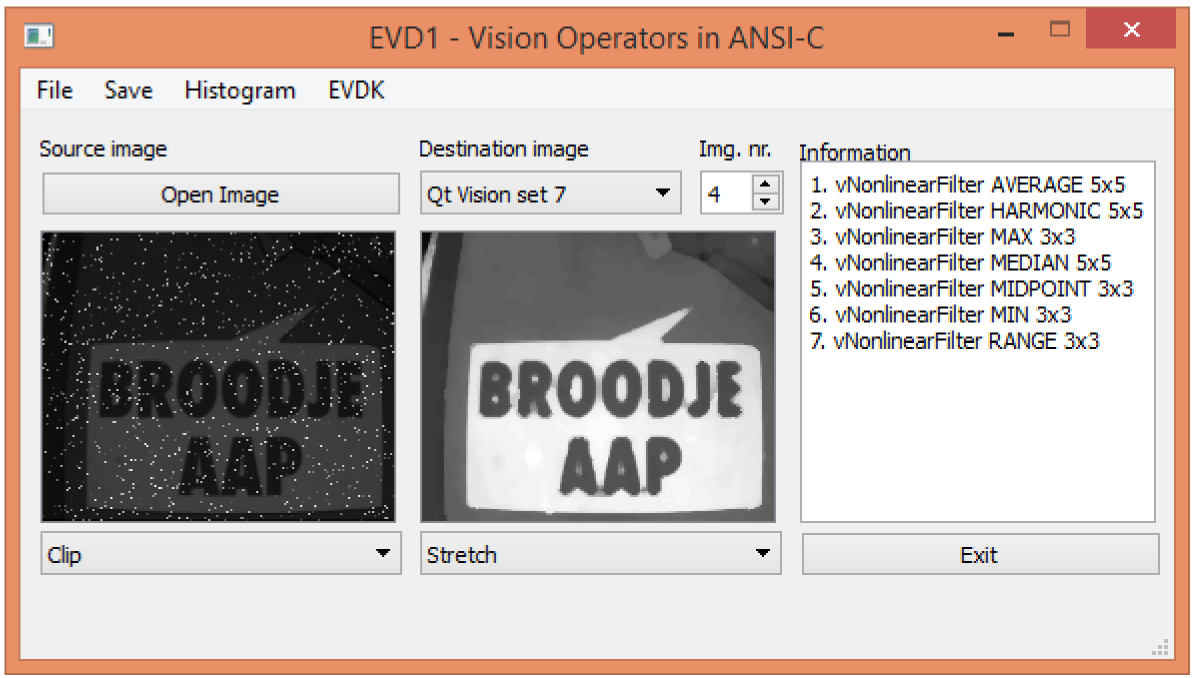
- **Used to remove outlier noise**
- **Used to remove <u>tailed</u> noise pixels**
- **Performs much better than the average filter, because edges are preserved**



Nonlinear filters: Median

Median filter:

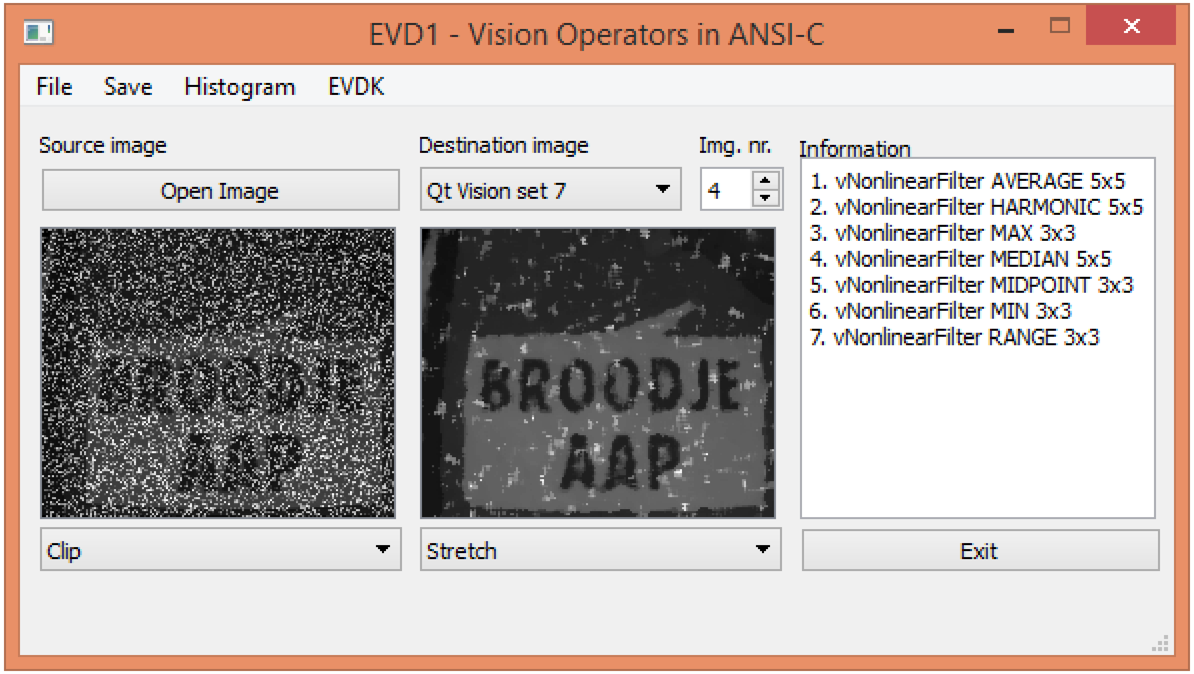
- **Used to remove outlier noise**
- **Used to remove <u>tailed</u> noise pixels**
- **Performs much better than the average filter, because edges are preserved**



Nonlinear filters: Median

Median filter:

- **Used to remove outlier noise**
- **Used to remove tailed noise pixels**
- **Performs much better than the average filter, because edges are preserved**



Nonlinear filters: Range

Rangefilter: Calculates the difference of the maximum and minimum pixel value within the window

	0					
0	1	1	3	1	…	…
	2	8	8	2	…	…
	2	1	3	1	…	…
	0	2	2	1	…	…

$$p_{(1,1)} = 9 - 1 = 8$$

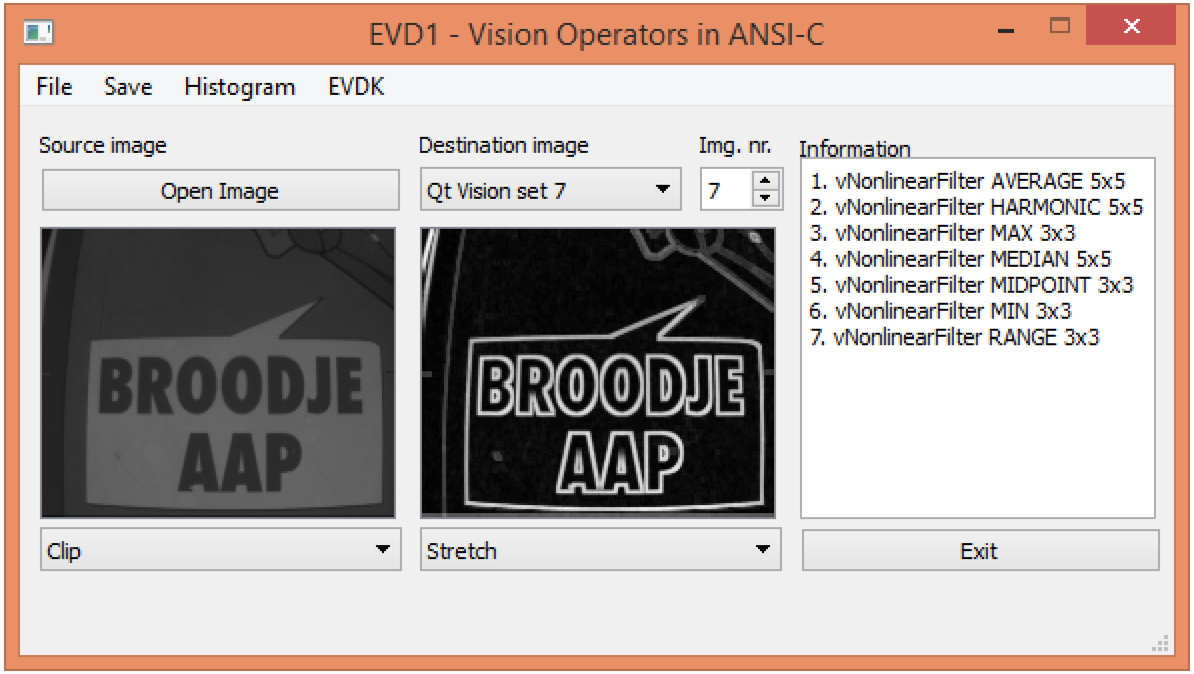
$$p_{(1,2)} = 9 - 1 = 8$$

$$p_{(y,x)} = max\,[window(y+i, x+j)]_{i=-^{n}/_{2}\; j=-^{n}/_{2}}^{i=^{n}/_{2}\; j=^{n}/_{2}} - min\,[window(y+i, x+j)]_{i=-^{n}/_{2}\; j=-^{n}/_{2}}^{i=^{n}/_{2}\; j=^{n}/_{2}}$$

Nonlinear filters: Range

Range filter:

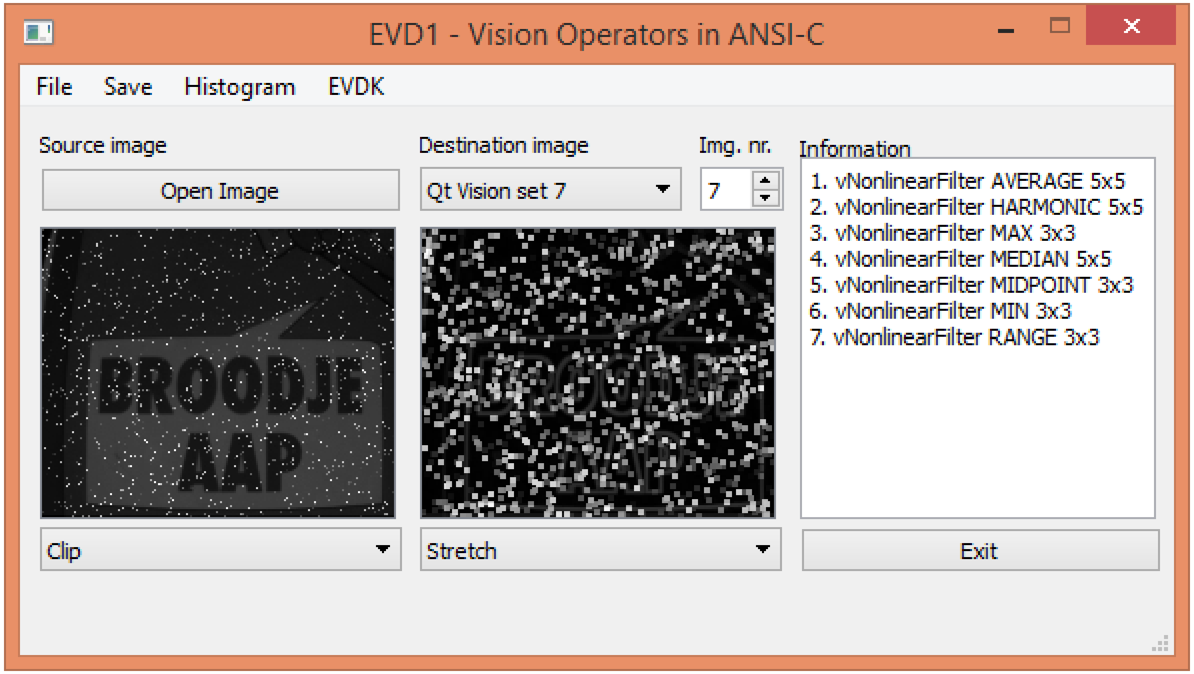
- **Used to find edges (if there is no noise)**



Nonlinear filters: Range

Range filter:

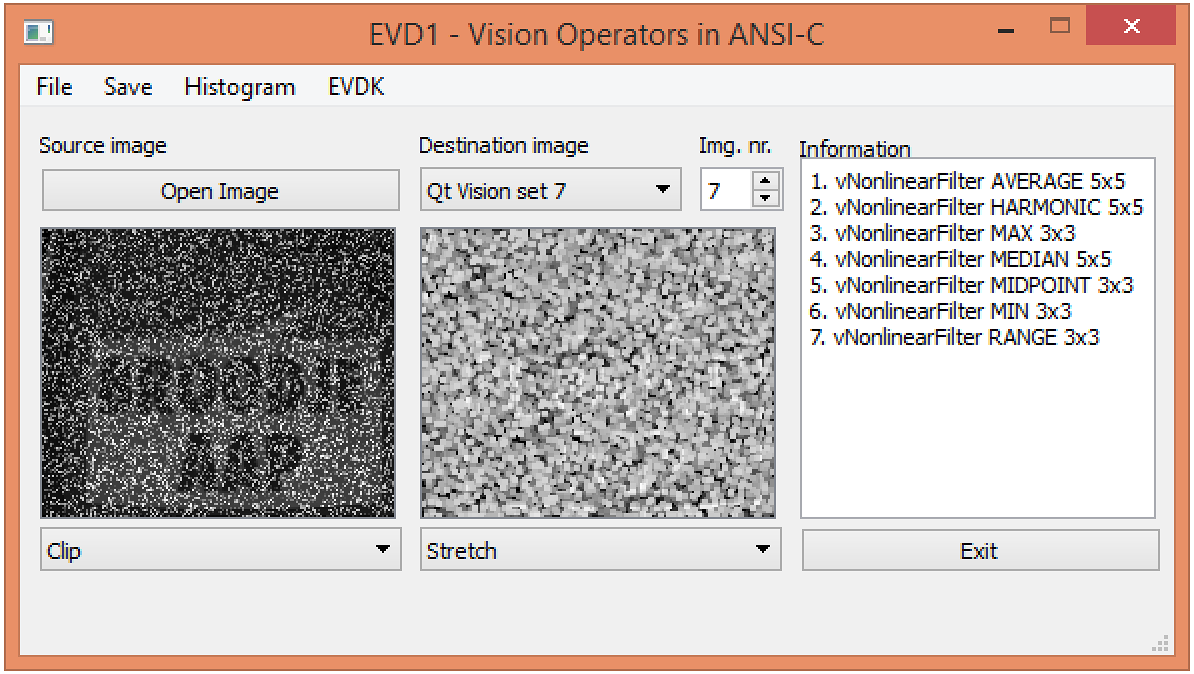
- **Used to find edges (if there is no noise)**



Nonlinear filters: Range

Range filter:

- **Used to find edges (if there is no noise)**



Nonlinear filters

How to handle edge pixels?

1	*1*	*1*	*3*	*1*		
1	1	1	3	1	…	…
2	2	3	9	2	…	…
2	2	1	3	1	…	…
	0	2	2	1	…	…

1. Do not handle them at all!

2. Only use available pixels

3. Extend edges

Example:

$$Average\ p_{(0,0)} = \frac{1}{4} \times (1 + 1 + 2 + 3) = 1{,}75$$

Nonlinear filters

Opdracht

Implement the function vNonlinearFilter()

Aangezien we met een variabel window moeten kunnen werken is het erg reken-intensief om ervoor te zorgen dat het src image ook de dst moet kunnen zijn.

Alléén voor deze operator: src ≠ dst

Dat betekent dat op de target deze operator als eerste aangeroepen moet worden!

Nonlinear filters

Opdracht

Geef in je logboek een verklaring voor alle filters waarom het filter juist wel of geen goed resultaat geeft bij het 'filters' plaatje met veel ruis.