Image Filtering in Python and OpenCV

PyGDA #13

Szymon
Moliński

Image Filtering

- bilateralFilter
- blur
- boxFilter
- buildPyramid
- dilate
- erode
- filter2D
- GaussianBlur
- getDerivKernels
- getGaussianKernel
- getGaborKernel
- getStructuringElement
- medianBlur
- morphologyEx
- Laplacian
- pyrDown
- pyrUp
- pyrMeanShiftFiltering
- sepFilter2D
- Smooth
- Sobel
- Scharr

AVERAGE (MEAN) FILTER

KERNELS:

$\bullet = \frac{1}{25}$

3×3: $\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$

5×5: $\begin{bmatrix} \frac{1}{25} & \bullet & \bullet & \bullet & \bullet \\ \frac{1}{25} & \bullet & \bullet & \bullet & \bullet \\ \frac{1}{25} & \bullet & \bullet & \bullet & \bullet \\ \frac{1}{25} & \bullet & \bullet & \bullet & \bullet \\ \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} & \frac{1}{25} \end{bmatrix}$

1. KERNEL SIZE: ODD NUMBER
2. SUM OF KERNEL MATRIX ELEMENTS => (1)
3. ALL ELEMENTS MUST BE THE SAME*

* BUT NOT ALWAYS ------> WEIGHTED MEAN FILTER

$\begin{bmatrix} \frac{1}{10} & \frac{1}{10} & \frac{1}{10} \\ \frac{1}{10} & \frac{1}{5} & \frac{1}{10} \\ \frac{1}{10} & \frac{1}{10} & \frac{1}{10} \end{bmatrix}$

Szymon Moliński

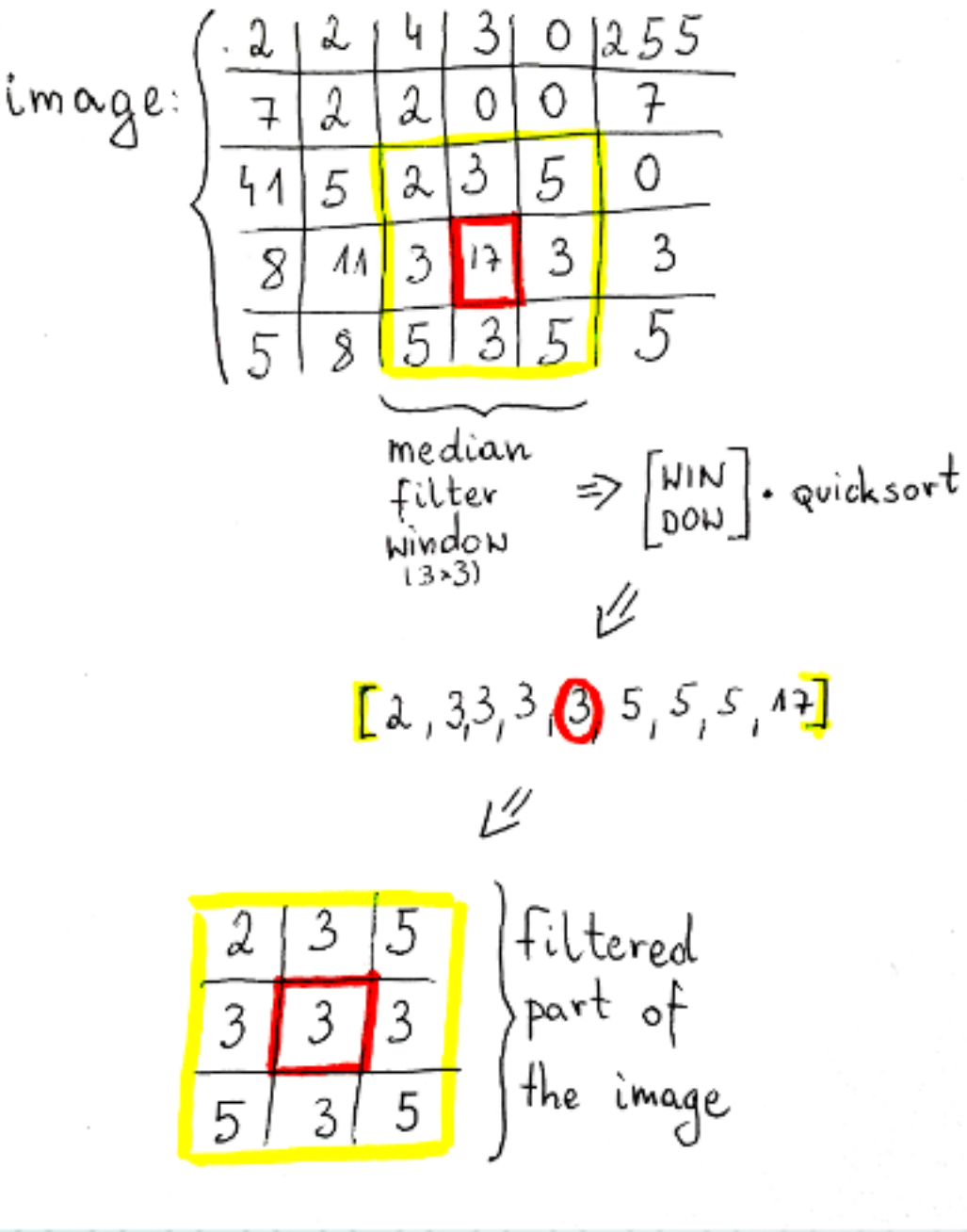
Average (mean) Filter

```
IMG = cv2.blur(IMG, sizeOfKernel)
```

http://homepages.inf.ed.ac.uk/rbf/HIPR2/mean.htm

Szymon Moliński

MEDIAN FILTER



Szymon Moliński

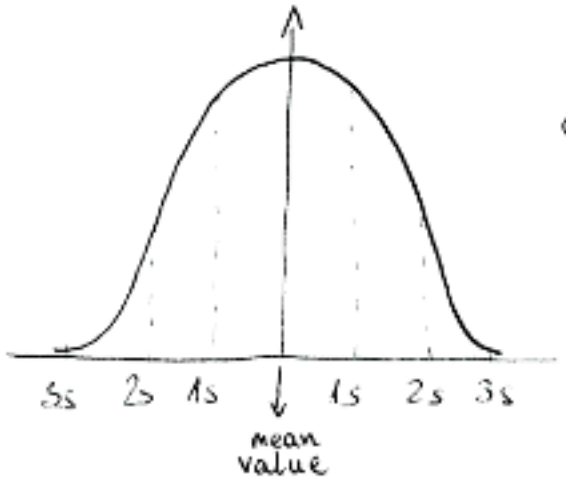
MEDIAN FILTER

```
IMG = cv2.medianBlur(IMG, sizeOfKernel)
```

http://homepages.inf.ed.ac.uk/rbf/HIPR2/median.htm

GAUSSIAN BLUR

KERNEL => BASED ON THE GAUSSIAN DISTRIBUTION



IMPULSE RESPONSE OF THE TYPICAL GAUSSIAN FILTER

Hl_1 = base matrix dependent on the kernel size
Hl_2 = -||- , but transposed
} 2D FILTER

s = sigma value

IMPLEMENTATION (EXAMPLE):

$$KERNEL = e^{\left(-(Hl_1^2 + Hl_2^2)/(2 \cdot s^2)\right)}$$

$$NORM\_KERNEL = \frac{KERNEL_{i,j}}{\sum KERNEL} \Rightarrow \frac{\text{each KERNEL element}}{\text{sum of all elements in KERNEL}}$$

PyGDA #13

Szymon Moliński

EXAMPLE OF GAUSSIAN BLUR KERNEL CREATION:

$s = 1;$
kernel size = 3×3

$$\Rightarrow \quad Hl_1 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$-1^2 + -1^2 = 2$
$1^2 + 0^2 = 1$
$0^2 + 0^2 = 0$

$$Hl_2 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$KERNEL = e^{(-(Hl_1^2 + Hl_2^2)/2)}$$

$$KERNEL = \begin{bmatrix} e^{-1} & e^{-0,5} & e^{-1} \\ e^{-0,5} & e^{0} & e^{-0,5} \\ e^{-1} & e^{-0,5} & e^{-1} \end{bmatrix}$$

$$KERNEL = \begin{bmatrix} 0,368 & 0,607 & 0,368 \\ 0,607 & 1 & 0,607 \\ 0,368 & 0,607 & 0,368 \end{bmatrix}$$

$$NORM\_KERNEL = \frac{KERNEL_{i,j}}{\Sigma(KERNEL)} = \frac{KERNEL_{i,j}}{4,898}$$

$$NORM\_KERNEL = \begin{bmatrix} 0,075 & 0,124 & 0,075 \\ 0,124 & 0,2 & 0,124 \\ 0,075 & 0,124 & 0,075 \end{bmatrix}$$

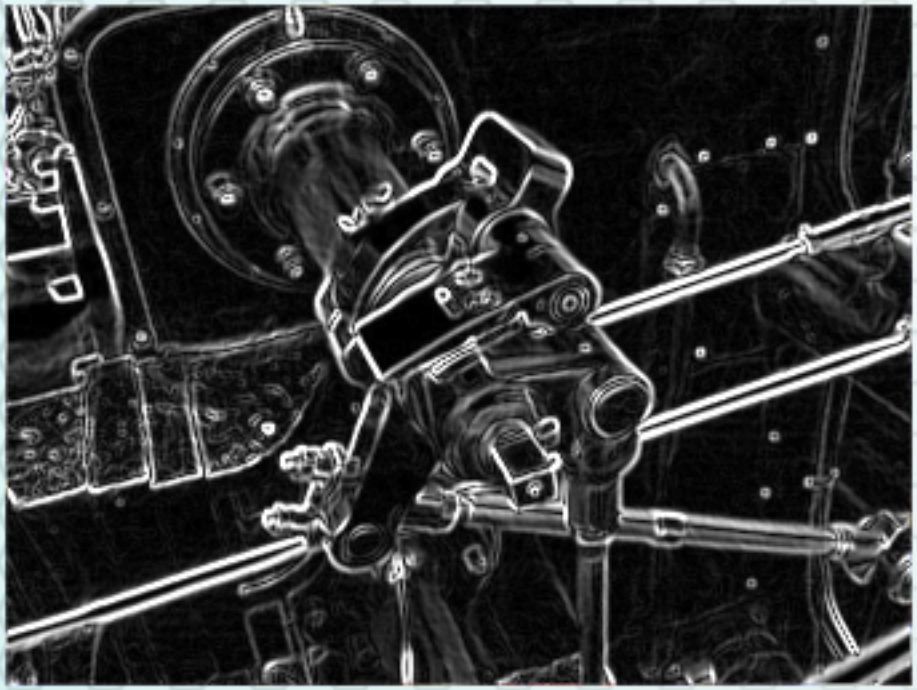
GAUSSIAN BLUR

PyGDA #13

Szymon Moliński

GAUSSIAN BLUR

```
IMG = cv2.GaussianBlur(IMG, sizeOfKernel, sigmax)
```



https://en.wikipedia.org/wiki/Gaussian_blur

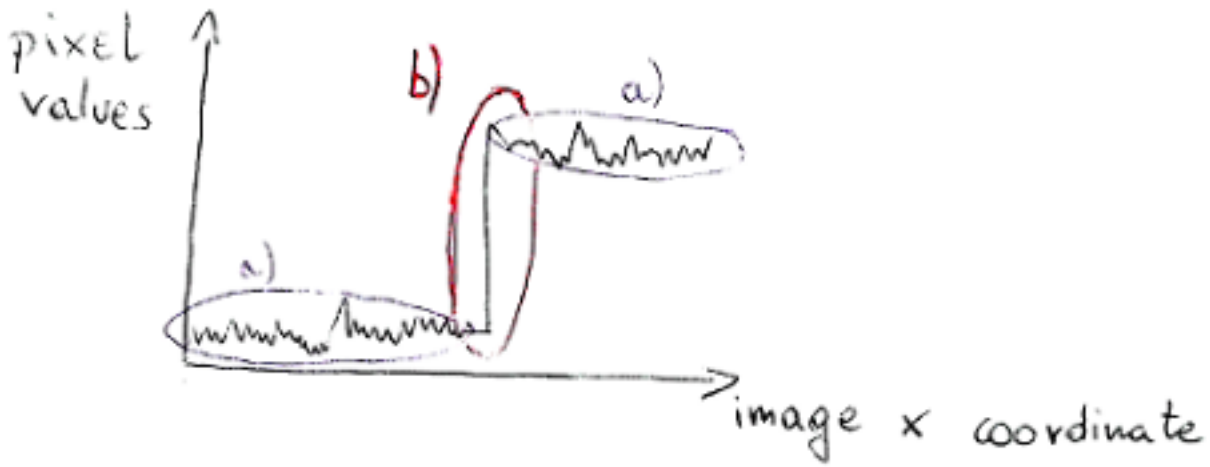
http://stackoverflow.com/questions/8204645/implementing-gaussian-blur-how-to-calculate-convolution-matrix-kernel

http://homepages.inf.ed.ac.uk/rbf/HIPR2/log.htm

BILATERAL FILTERING

REMOVES NOISE BUT ALSO PRESERVES THE EDGES

EXAMPLE:

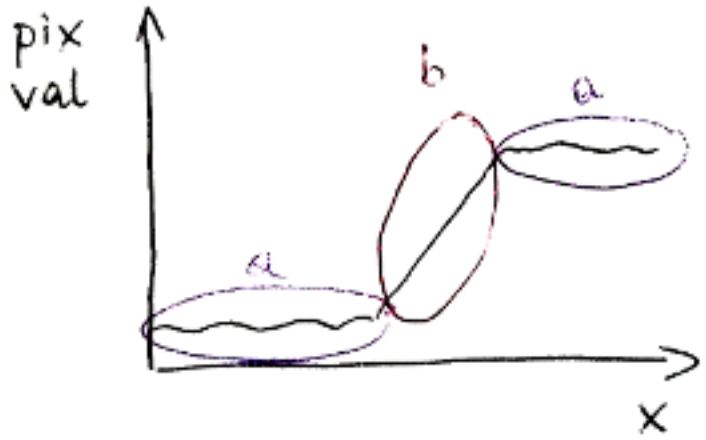


a) → noise region

b) → edge region

Szymon Moliński

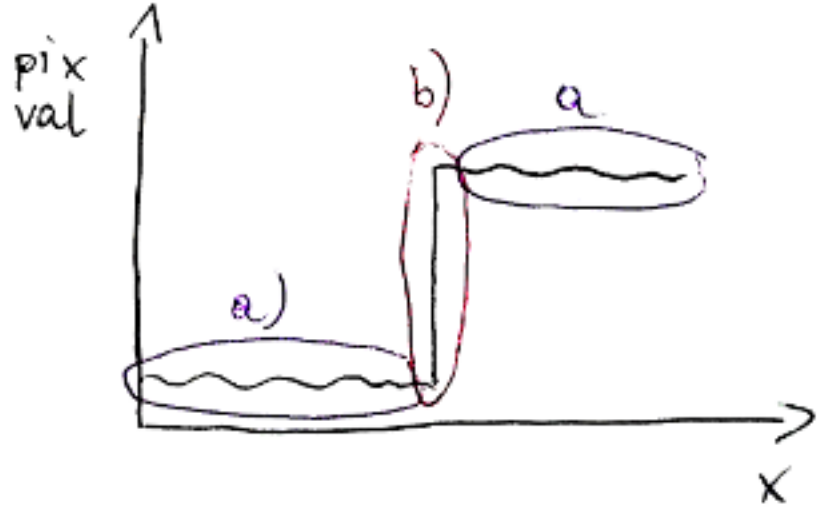
MEAN or GAUSSIAN



a) → reduced noise

b) → blurred edge

BILATERAL



a) → reduced noise

b) → preserved edge

BILATERAL FILTERING

BILATERAL FILTERING

```
cv2.bilateralFilter(src, d, sigmaColor, sigmaSpace)
```

http://people.csail.mit.edu/sparis/bf_course/

A
B
C

DILATION ↘
EROSION ↗ MORPHOLOGICAL OPERATORS

cv2.dilate(src, kernel)
cv2.erode(src, kernel)

http://homepages.inf.ed.ac.uk/rbf/HIPR2/morops.htm

BONUS EXAMPLE
Converting classical photo to animated version
(look also to):

http://stackoverflow.com/questions/1357403/how-to-cartoon-ify-an-image-programmatically

CHECK CODE ON GITHUB:

https://github.com/szymonMolinski/PyGDA13

Szymon's E-mail:

s.molinski@protonmail.ch

Szymon's blog:

imgsimon.blogspot.com