Operacja splotu (ang. convolution)

Artur Gramacki
a.gramacki@issi.uz.zgora.pl
Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych
2020

Definicje

- Ciągła

$$(x * h)(t) = y(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau$$

Gwiazdka to zwyczajowy symbol operacji splotu

- Dyskretna

$$(x * h)[i] = y[i] = \sum_{j=-\infty}^{j=+\infty} x[j] \cdot h[i-j]$$

Kropeczki zwykle nie piszemy, tu ją postawiliśmy dla podkreślenia, że oba sygnały mnożymy przez siebie

- Dyskretna, realizowalne fizycznie

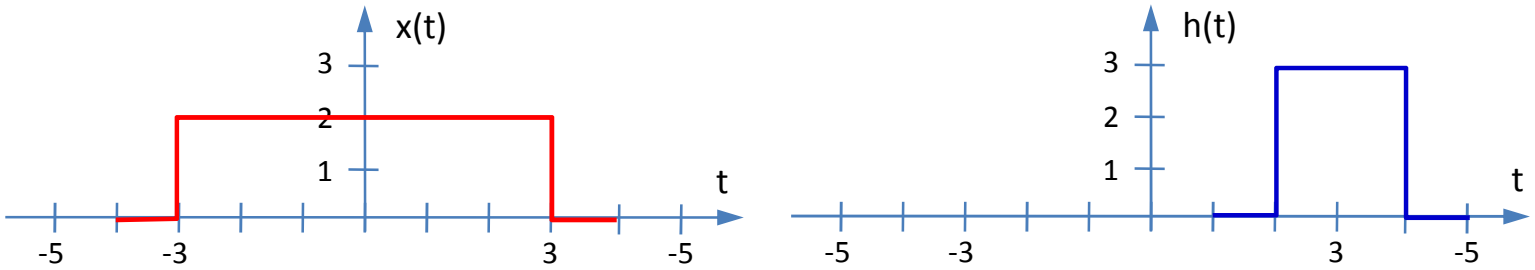
$$(x * h)[i] = y[i] = \sum_{j=0}^{M-1} x[j] \cdot h[i-j]$$

Nawiasy kwadratowe zamiast okrągłych przypominają, że operujemy w dziedzinie czasu dyskretnego

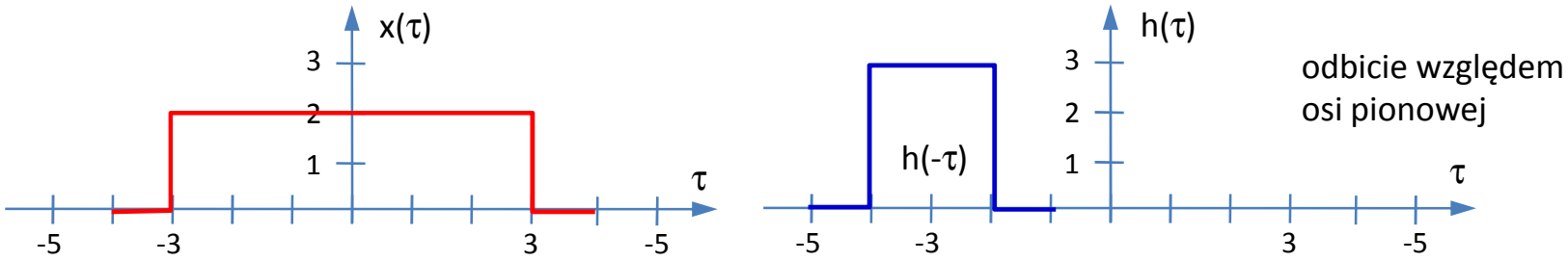
Definiuje się też splot cykliczny, którego my jednak nie używamy

Prosty przykład, „ręczne całkowanie”

- Dwa sygnały: x(t) oraz h(t)



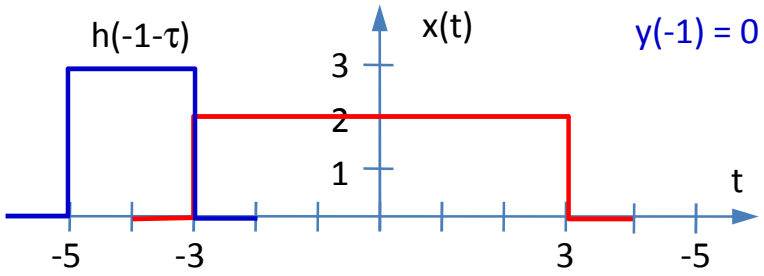
- x(t), h(t) jako funkcje zmiennej τ, rysujemy przbieg h(-τ)

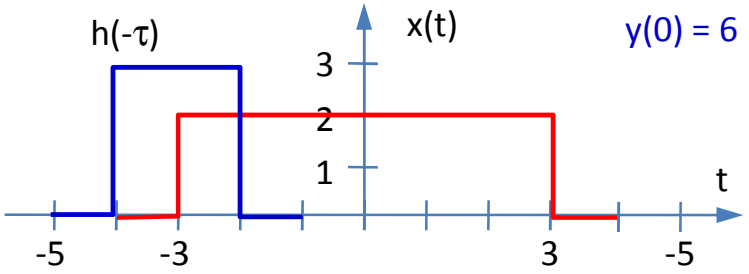


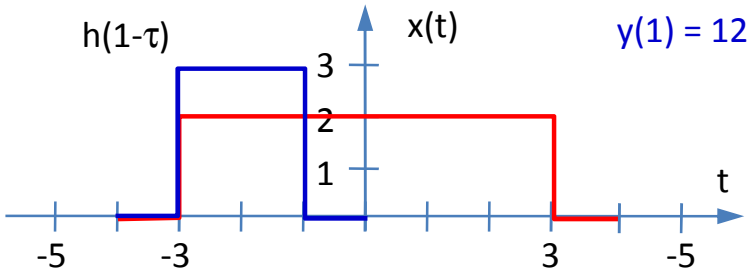
$$(x * h)(t) = y(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau$$

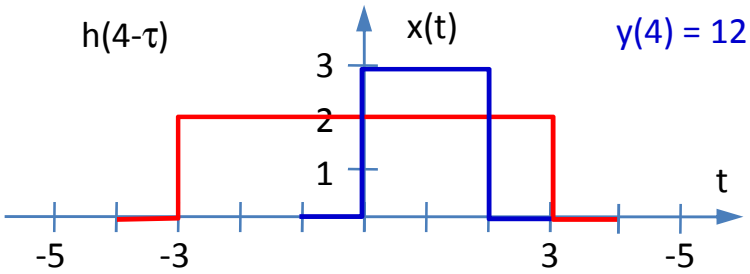
Prosty przykład, „ręczne całkowanie”

- Rysujemy sygnał h(t- τ) dla różnych chwil czasowych t, licząc za każdym razem pole powierzchni pod wykresem iloczynu x(τ) oraz h(t- τ)





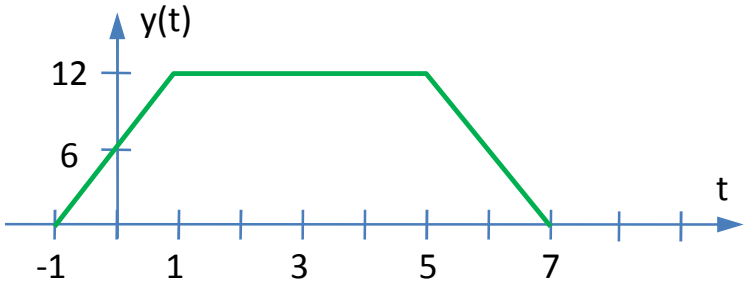




$$(x * h)(t) = y(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau$$

Prosty przykład, „ręczne całkowanie”

- Wykres finalny y(t)



- Po klasycznym wyliczeniu całki otrzymamy następujące rozwiązanie

$$y(t)=\begin{cases}0 \quad dla \quad t\le -1, t\ge 7\\ 6t+6 \quad dla \quad -1<t\le 1\\ 12 \quad dla \quad 1<t\le 5\\ 6t+12 \quad dla \quad 5<t\le 7\end{cases}$$

- W dziedzinie DSP oraz Vision sygnały ZAWSZE są dyskretne

Prosty przykład numeryczny

- Wektor x ma 4 elementy a wektor h 2 elementy, jak niżej
  - elementy wektora x mają indeksy 0 … M - 1, czyli (0, 1, 2, 3)
  - elementy wektora h mają indeksy 0 … N - 1, czyli (0, 1)
  - rozmiar wyniku to M + N - 1

$$x = [1 \quad 3 \quad -2 \quad 1]$$

$$h = [1 \quad 2]$$

$$M = 4$$

$$N = 2$$

$$j = 0 \ldots 3$$

$$i = 0 \ldots 4 \quad (i = 0 \ldots M + N - 2)$$

$$y[i] = \sum_{j=0}^{M-1} x[j] \cdot h[i - j]$$

Prosty przykład numeryczny

- Obliczenia

$$y[0] = x[0] \cdot h[0] + x[1] \cdot h[-1] + x[2]vh[-2] + x[3] \cdot h[-3] = 1 + 0 + 0 + 0 = 1$$

$$y[1] = x[0] \cdot h[1] + x[1] \cdot h[0] + x[2] \cdot h[-1] + x[3] \cdot h[-2] = 2 + 3 + 0 + 0 = 5$$

$$y[2] = x[0] \cdot h[2] + x[1] \cdot h[1] + x[2] \cdot h[0] + x[3] \cdot h[-1] = 0 + 6 - 2 + 0 = 4$$

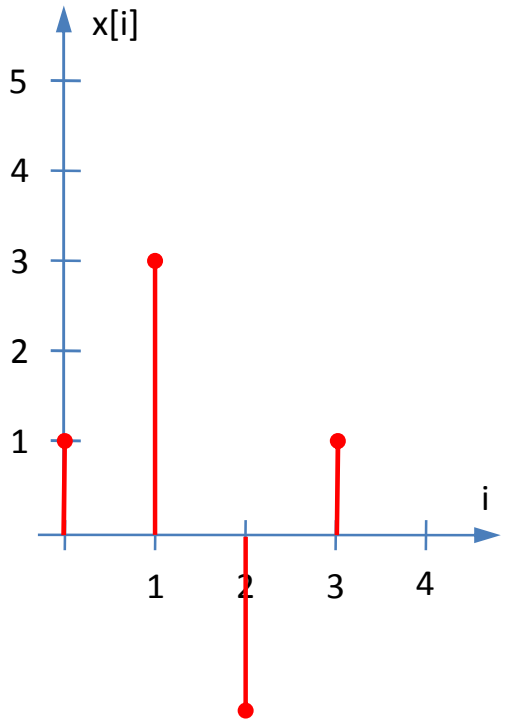
$$y[3] = x[0] \cdot h[3] + x[1] \cdot h[2] + x[2] \cdot h[1] + x[3] \cdot h[0] = 0 + 0 - 4 + 1 = -3$$

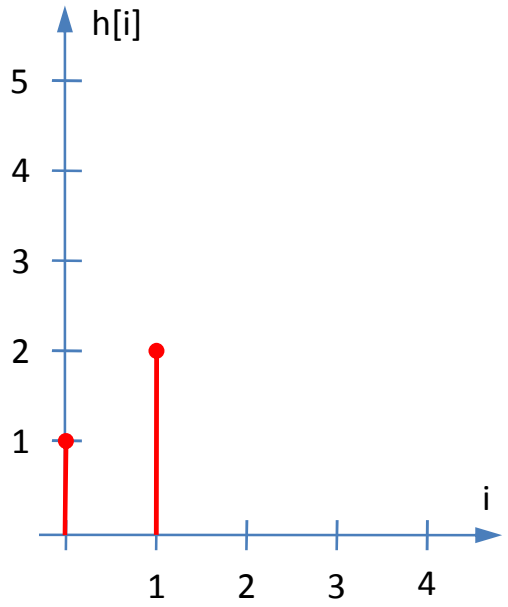
$$y[4] = x[0] \cdot h[4] + x[1] \cdot h[3] + x[2] \cdot h[2] + x[3] \cdot h[1] = 0 + 0 + 0 + 2 = 2$$

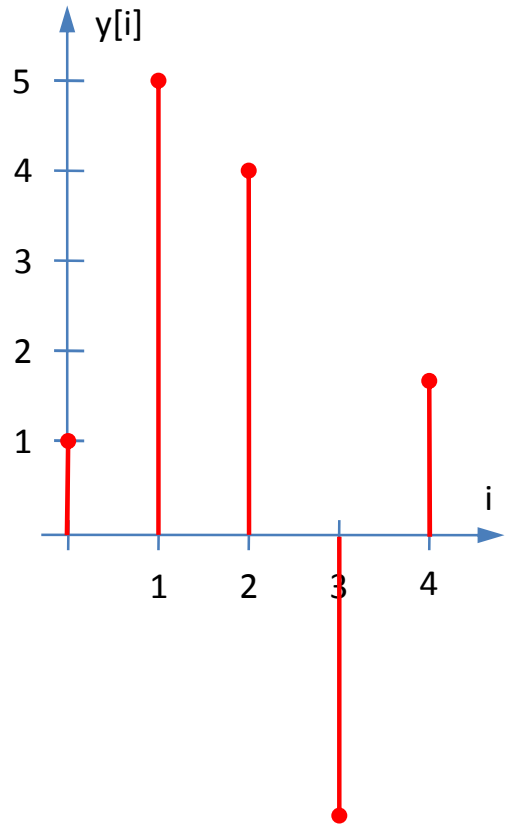
$$y[i] = \sum_{j=0}^{M-1} x[j] \cdot h[i - j]$$

Prosty przykład numeryczny

- Wektory x[i], h[i], y[i]



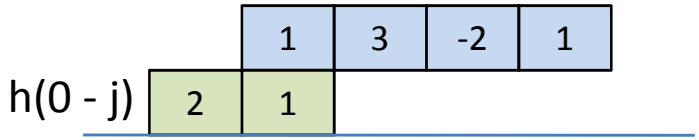




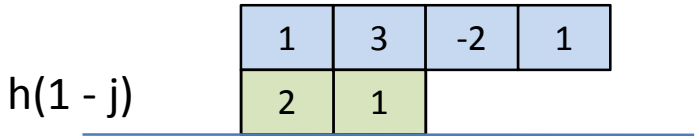
$x = [1 \quad 3 \quad -2 \quad 1]$ $h = [1 \quad 2]$ $y = [1 \quad 5 \quad 4 \quad -3 \quad 2]$

Prosty przykład numeryczny

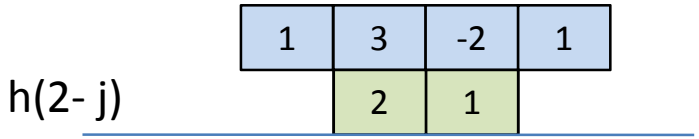
- Obliczenia zwizualizowane, y = x ∗ h



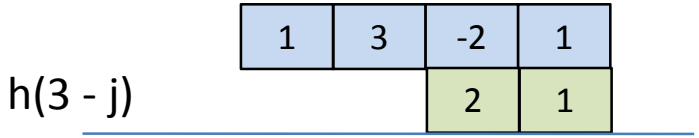
y[0] = 1·1 = 1



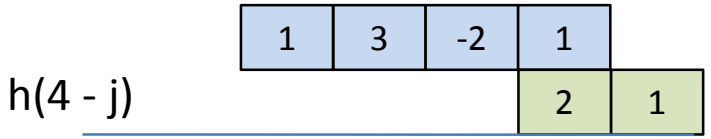
y[1] = 1·2 + 1·3 = 5



y[2] = 2·3 - 2·1 = 4

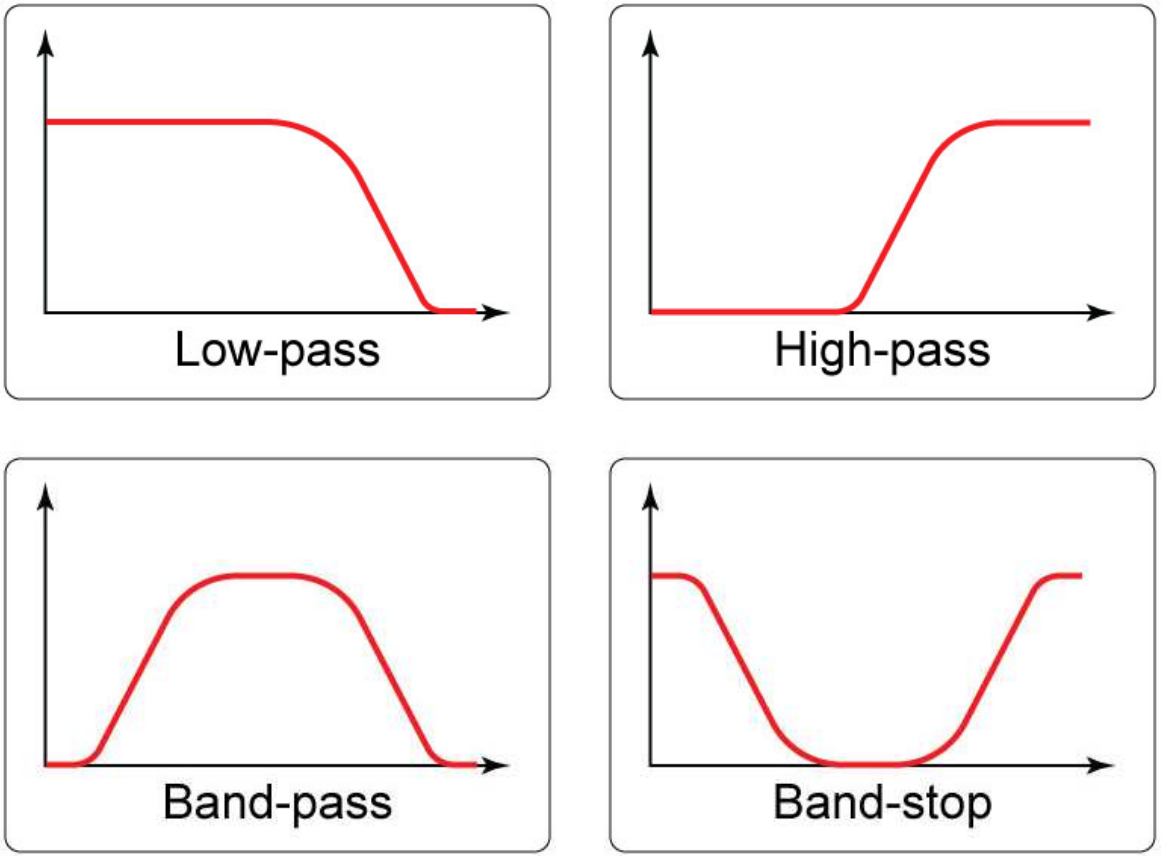


y[3] = -2·2 + 1·1 = -3



y[4] = 1·2 = 2

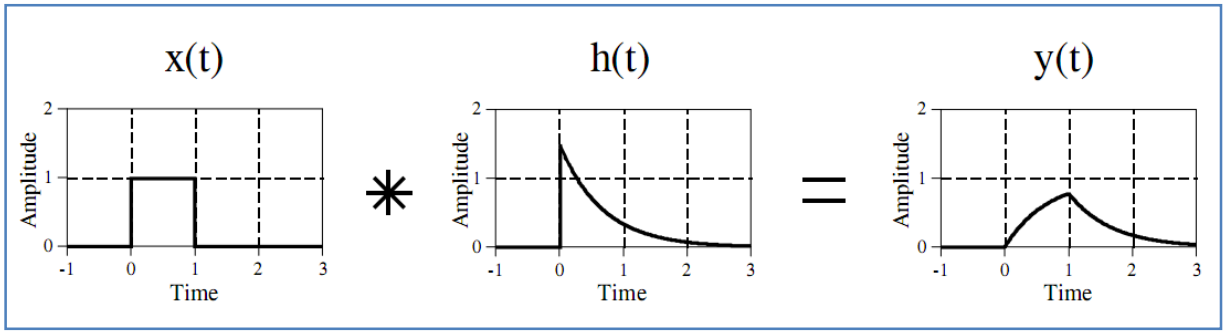
Splot 1D, signal processing

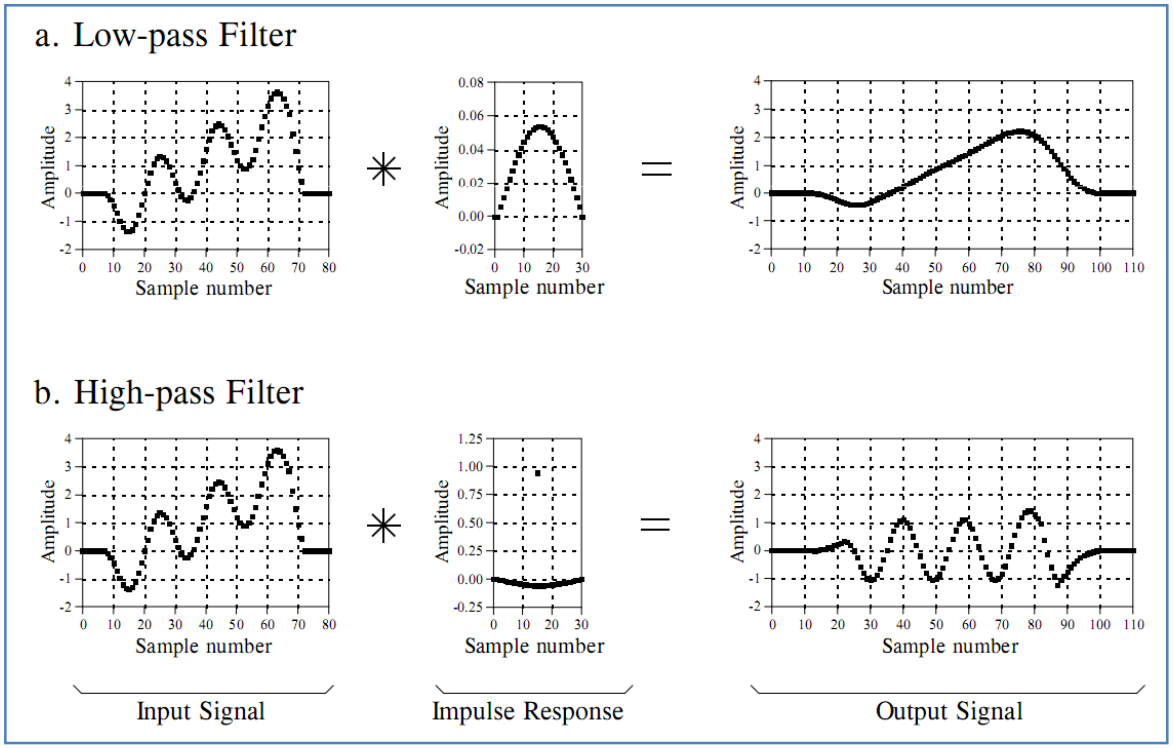


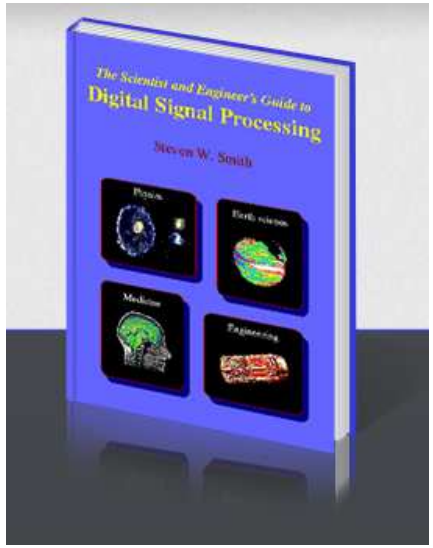
źródło: https://www.allaboutcircuits.com/technical-articles/low-pass-filter-tutorial-basics-passive-RC-filter/

Splot 1D, signal processing

$$y(t) = \int_{-\infty}^{+\infty} x(\tau)\, h(t-\tau)\, d\tau$$

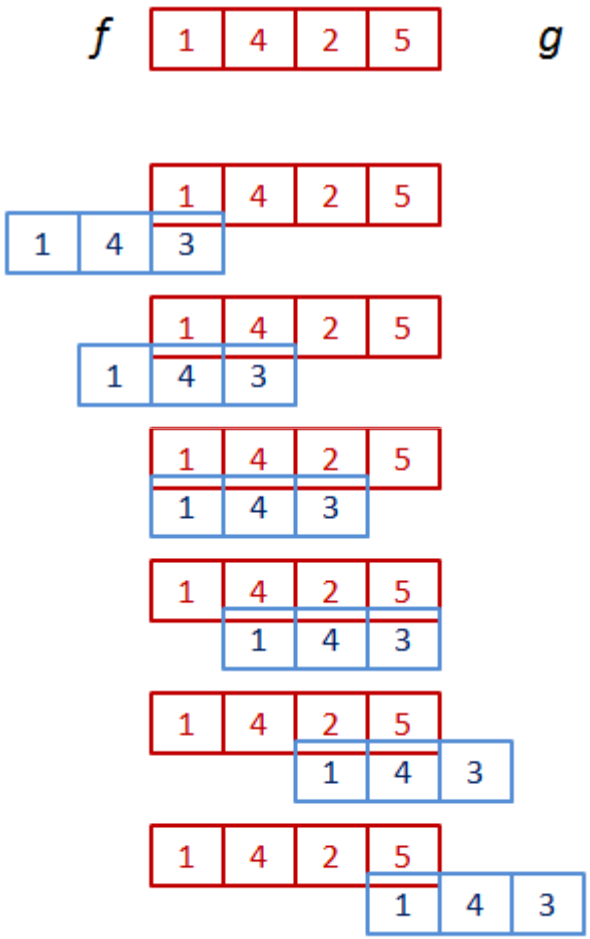






źródło: https://www.dspguide.com

Splot 1D, wizualizacja



C[0] = 1*3 = **3**

C[1] = 1*4 + 4*3 = **16**

C[2] = 1*1 + 4*4 + 2*3 = **23**

C[3] = 4*1 + 2*4 + 5*3 = **27**

C[4] = 2*1 + 5*4 = **22**

C[5] = 5*1 = **5**

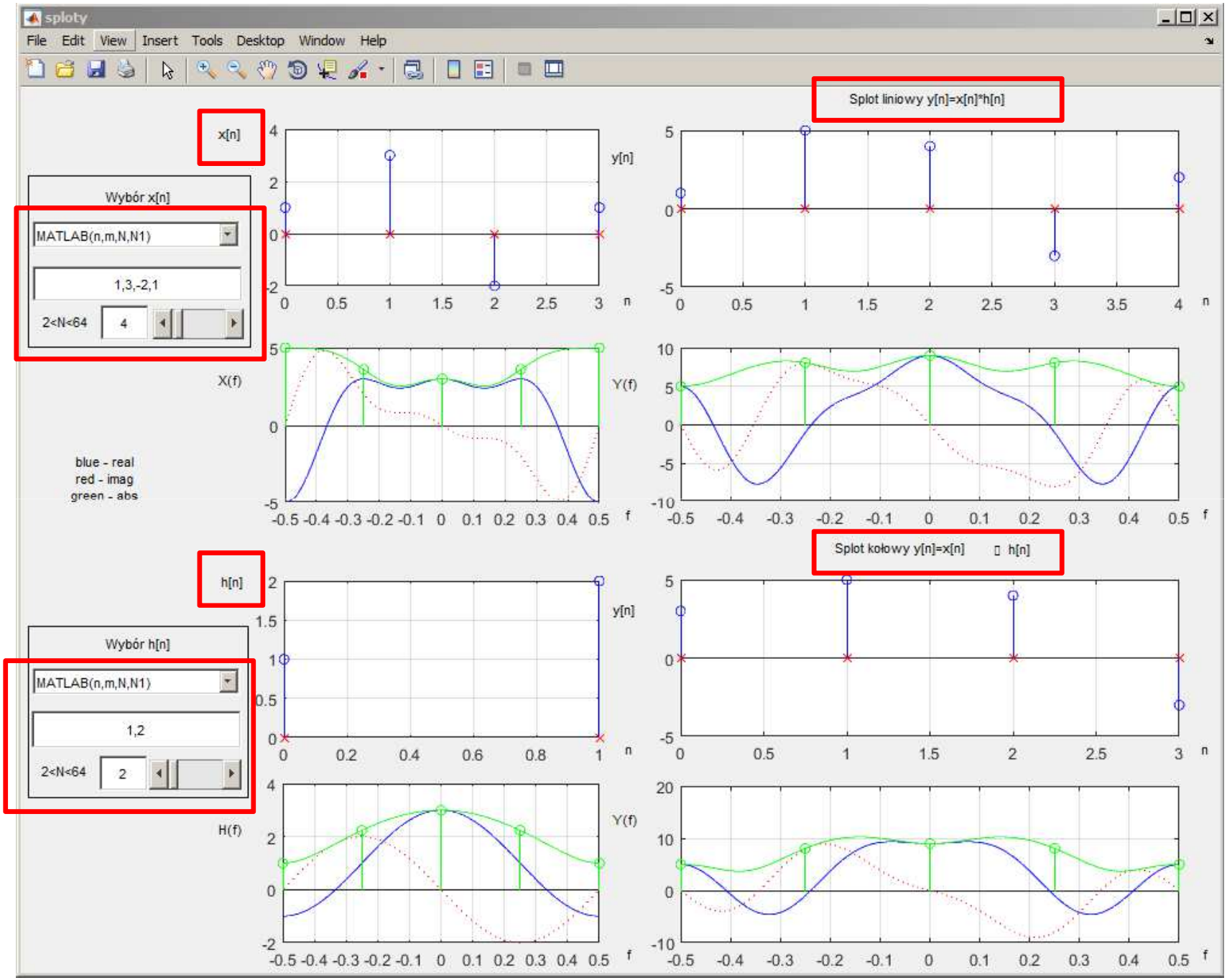
http://toto-share.com

Splot 1D, przykłady numeryczne

```
x = [1, 3, -2, 1]  % M = 4
h = [1, 2]         % N = 2
conv(x, h)         % długość wyniki M + N – 1 = 5
ans =
     1   5   4  -3   2

% mogę sobie dowolnie zerami pouzupełniać
x1 = [0, 0, 1, 3, -2, 1, 0, 0, 0]
h1 = [0, 0, 1, 2, 0, 0, 0]
conv(x1, h1)
ans =
     0   0   0   0   1   5   4  -3   2   0   0   0   0   0   0

% splot kołowy
cconv(x, h, 4)
ans =
     3   5   4   -3

% gdy N = LENGTH(x) + LENGTH(h) - 1 to conv = cconv
cconv(x, h, 5)
ans =
    1.0000   5.0000   4.0000  -3.0000   2.0000
```

Splot 1D, program w Matlab

- https://sound.eti.pg.gda.pl/~landos/Laboratorium_CPS/
  - bardzo funkcjonalne narzędzie dydaktyczne, wykonane w Matlab-ie
  - link jest już niestety niedostępny (wrzesień 2020)
  - autor, dr inż. Andrzej Leśnicki (Wydziału Elektroniki, Telekomunikacji i Informatyki Politechniki Gdańskiej) zmarł 19 marca 2018


sploty
File Edit View Insert Tools Desktop Window Help
x[n]
Wybór x[n]
MATLAB(n,m,N,N1)
1,3,-2,1
2<N<64
4
X(f)
blue - real
red - imag
green - abs
y[n]
n
Splot liniowy y[n]=x[n]*h[n]
Y(f)
f
h[n]
Wybór h[n]
MATLAB(n,m,N,N1)
1,2
2<N<64
2
H(f)
Splot kołowy y[n]=x[n] h[n]

Splot 2D

- Dla przypomniana definicja splote 1D

$$(x * h)[i] = y[i] = \sum_{j=0}^{M-1} x[j] \cdot h[i-j]$$

- Definicja (dyskretna) splotu 2D

$$(x * h)[n_1, n_2] = y[n_1, n_2] = \sum_{k_1=0}^{M_1-1} \sum_{k_2=0}^{M_2-1} x[k_1, k_2] \cdot h[n_1 - k_1, n_2 - k_2]$$

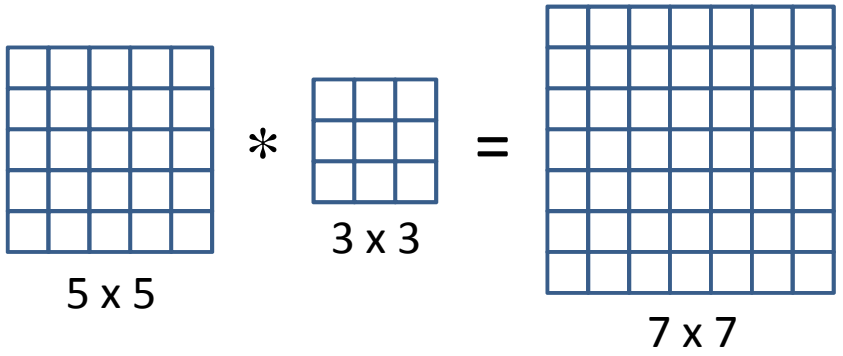
$y : N_1 \; x \; N_2$

$x : M_1 \; x \; M_2$

$h : P_1 \; x \; P_2$

$N_1 = M_1 + P_1 - 1$

$N_2 = M_2 + P_2 - 1$



Splot 2D

- Przykład obliczeń (wprost ze wzoru)

$$y[1,1] = x[0,0]\cdot h[1,1] + x[1,0]\cdot h[0,1] + x[2,0]\cdot h[-1,1] + x[0,1]\cdot h[1,0] + x[1,1]\cdot h[0,0] +$$
$$x[2,1]\cdot h[-1,0] + x[0,2]\cdot h[1,-1] + x[1,2]\cdot h[0,-1] + x[2,2]\cdot h[-1,-1]$$

1	3	1
0	-1	1
2	2	-1

input

$*$

1	2
0	-1

kernel

-1	0
2	-1

kernel po odbiciu w poziomie i w pionie

Aby łatwiej się liczyło, patrz niżej

$$\Rightarrow 1\cdot(-1) + 3\cdot 0 + 0\cdot 2 + 1\cdot(-1) = -2$$

$$\Rightarrow 3\cdot(-1) + 1\cdot 0 + (-1)\cdot 2 + 1\cdot 1 = -4$$

$$\Rightarrow 0\cdot(-1) + (-1)\cdot 0 + 2\cdot 2 + 2\cdot 1 = 6$$

$$\Rightarrow 1\cdot(-1) + 1\cdot 0 + 2\cdot 2 + (-1)\cdot 1 = 4$$

-2	-4
6	4

wynik

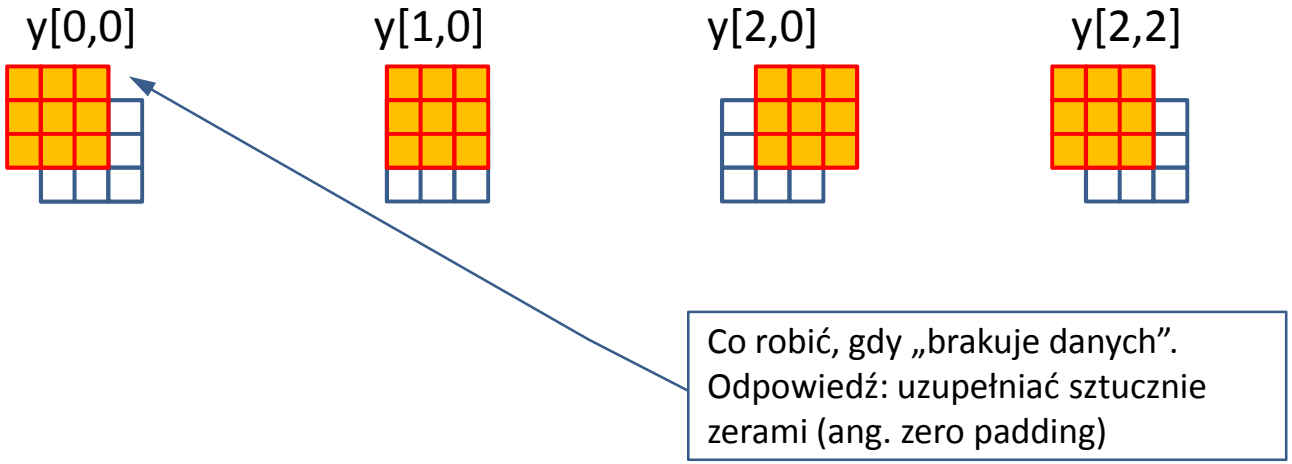
Splot 2D

- Wizualizacja obliczeń

$$x=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix},\quad h=\begin{bmatrix}-1 & -2 & -1\\ 0 & 0 & 0\\ 1 & 2 & 1\end{bmatrix},\quad y=\begin{bmatrix}-13 & -20 & -17\\ -18 & -24 & -18\\ 13 & 20 & 17\end{bmatrix}$$

$$h\_flipped=\begin{bmatrix}1 & 2 & 1\\ 0 & 0 & 0\\ -1 & -2 & -1\end{bmatrix}$$

kernel po odbiciu w poziomie i w pionie



Splot 2D, przykłady numeryczne

```
% splot 2D
x2 = [1 3 1; 0 -1 1; 2 2 -1]
h2 = [1 2; 0 -1]
x2 =
     1     3     1
     0    -1     1
     2     2    -1
h2 =
     1     2
     0    -1

conv2(x2, h2, 'full')
ans =
     1     5     7     2
     0    -2    -4     1
     2     6     4    -3
     0    -2    -2     1

conv2(x2, h2, 'same')
ans =
    -2    -4     1
     6     4    -3
    -2    -2     1

conv2(x2, h2, 'valid')
ans =
    -2    -4
     6     4
```

Splot 2D, przykłady numeryczne

X

```
[,1] [,2] [,3] [,4] [,5]
[1,]    4    4    3    5    4
[2,]    6    6    5    5    2
[3,]    5    6    6    6    2
[4,]    6    7    5    5    3
[5,]    3    5    2    4    4
```

h

```
     [,1] [,2] [,3]
[1,]    0   -1    0
[2,]   -1    4   -1
[3,]    0   -1    0
```

Y ('same')

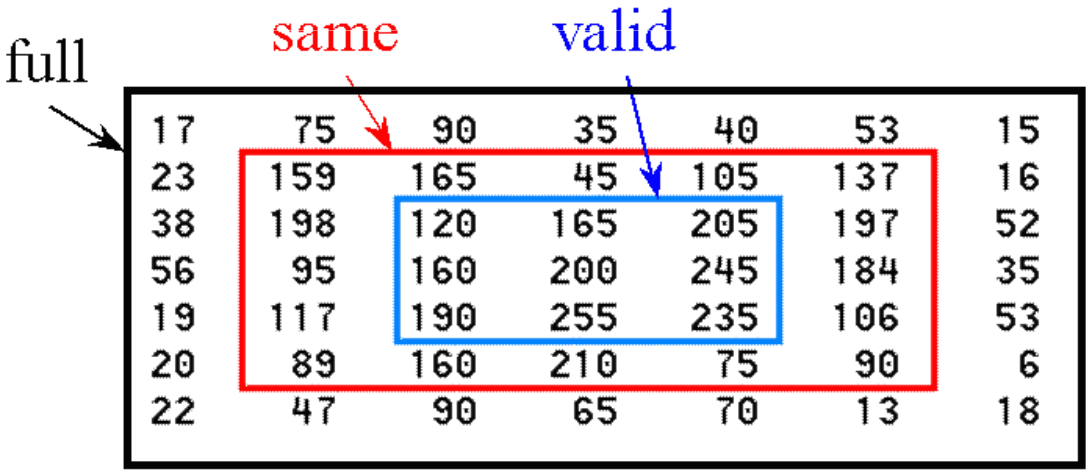
```
    [,1] [,2] [,3] [,4] [,5]
[1,]    6    3   -2    8    9
[2,]    9    3    0    2   -3
[3,]    2    0    2    6   -3
[4,]    9    6    0    2    1
[5,]    1    8   -6    5    9
```

Y (‘full')

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7]
[1,]    0   -4   -4   -3   -5   -4    0
[2,]   -4    6    3   -2    8    9   -4
[3,]   -6    9    3    0    2   -3   -2
[4,]   -5    2    0    2    6   -3   -2
[5,]   -6    9    6    0    2    1   -3
[6,]   -3    1    8   -6    5    9   -4
[7,]    0   -3   -5   -2   -4   -4    0
```

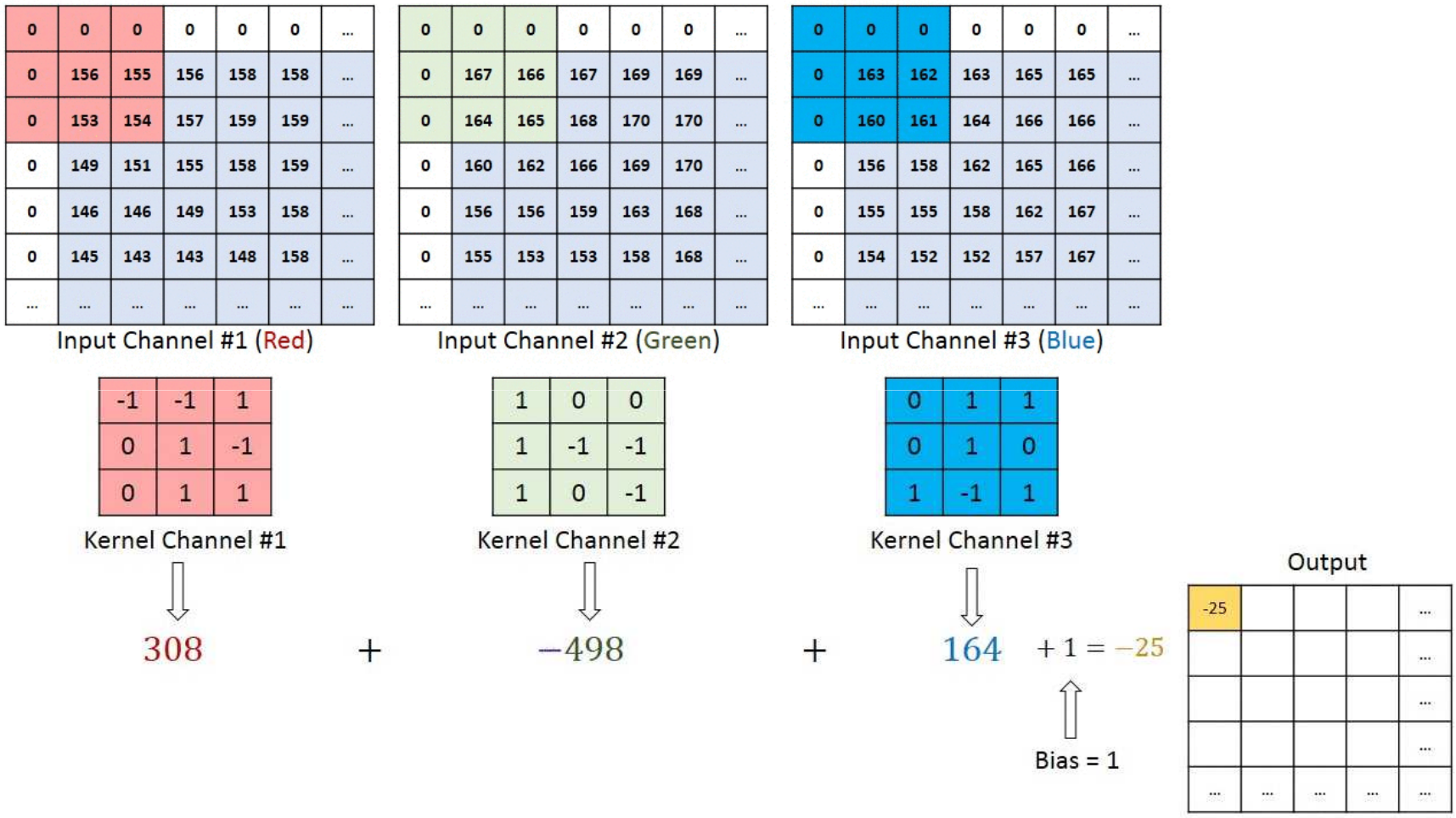
Splot 2D, przykłady numeryczne

$$\begin{pmatrix} 17 & 24 & 1 & 8 & 15 \\ 23 & 5 & 7 & 14 & 16 \\ 4 & 6 & 13 & 20 & 22 \\ 10 & 12 & 19 & 21 & 3 \\ 11 & 18 & 25 & 2 & 9 \end{pmatrix} * \begin{pmatrix} 1 & 3 & 1 \\ 0 & 5 & 0 \\ 2 & 1 & 2 \end{pmatrix}$$



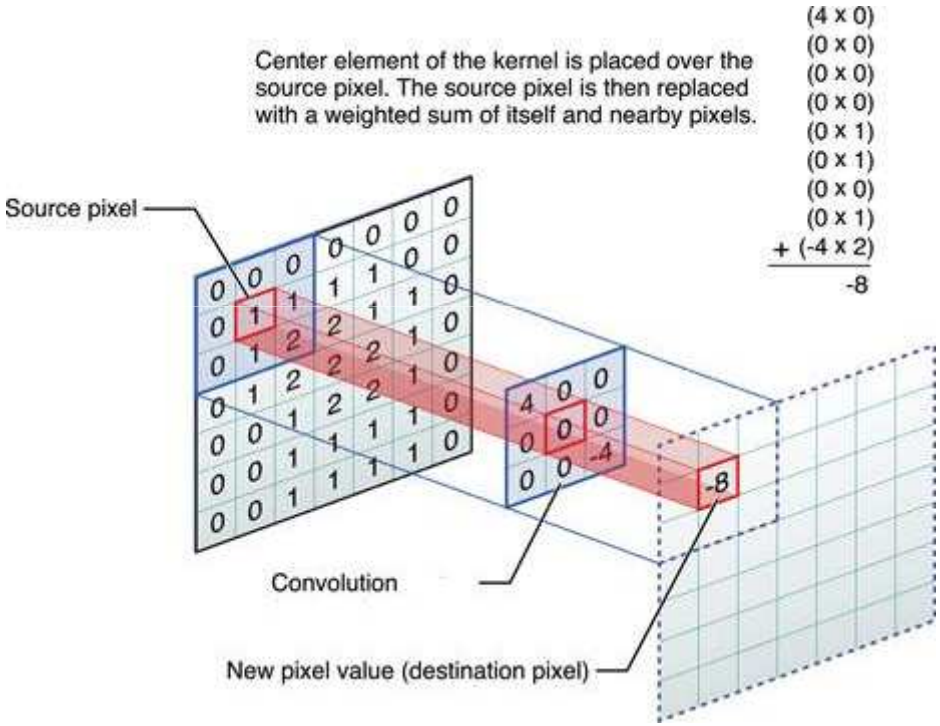
źródło: https://johnloomis.org/ece563/notes/filter/conv/convolution.html

Splot 2D, zero padding



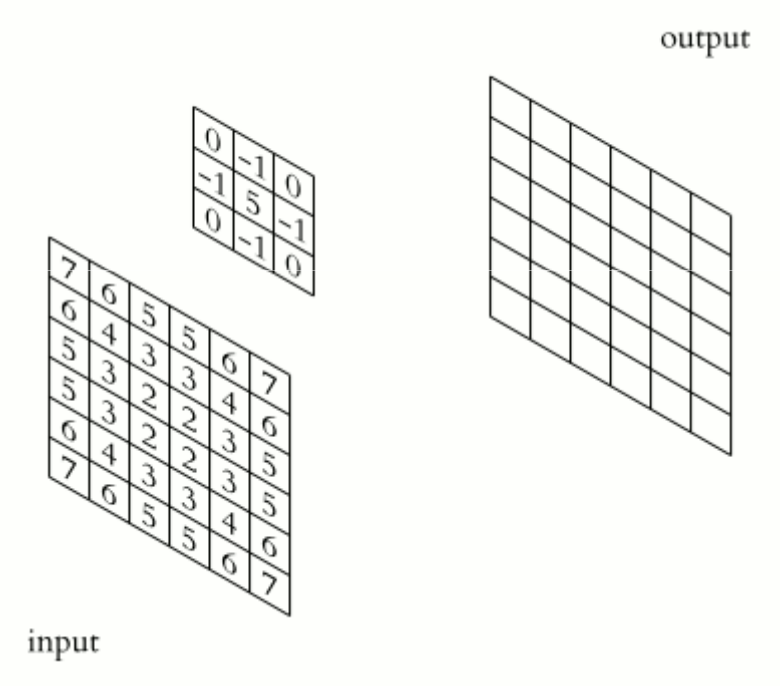
źródło: https://mmuratarat.github.io/2019-01-17/implementing-padding-schemes-of-tensorflow-in-python

Splot 2D, wizualizacje



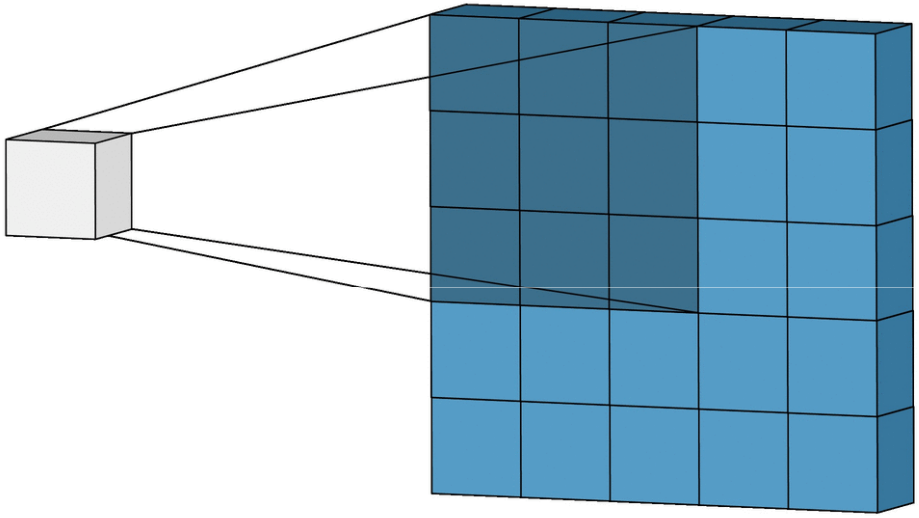
źródło: https://medium.com/@bdhuma/6-basic-things-to-know-about-convolution-daef5e1bc411

Splot 2D, wizualizacje



źródło: https://towardsdatascience.com/types-of-convolution-kernels-simplified-f040cb307c37

Splot 2D, wizualizacje



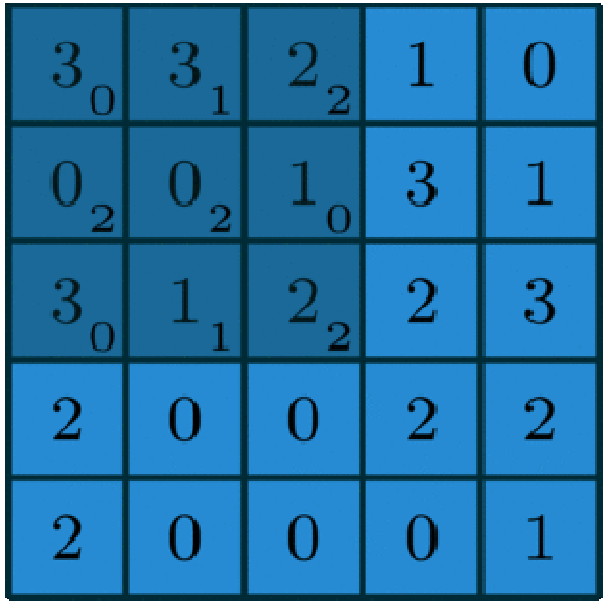
źródło: https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1

Splot 2D, wizualizacje

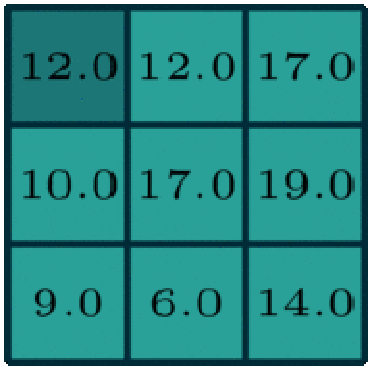
kernel (h)

0	1	2
2	2	0
0	1	2

dane wejściowe (x)

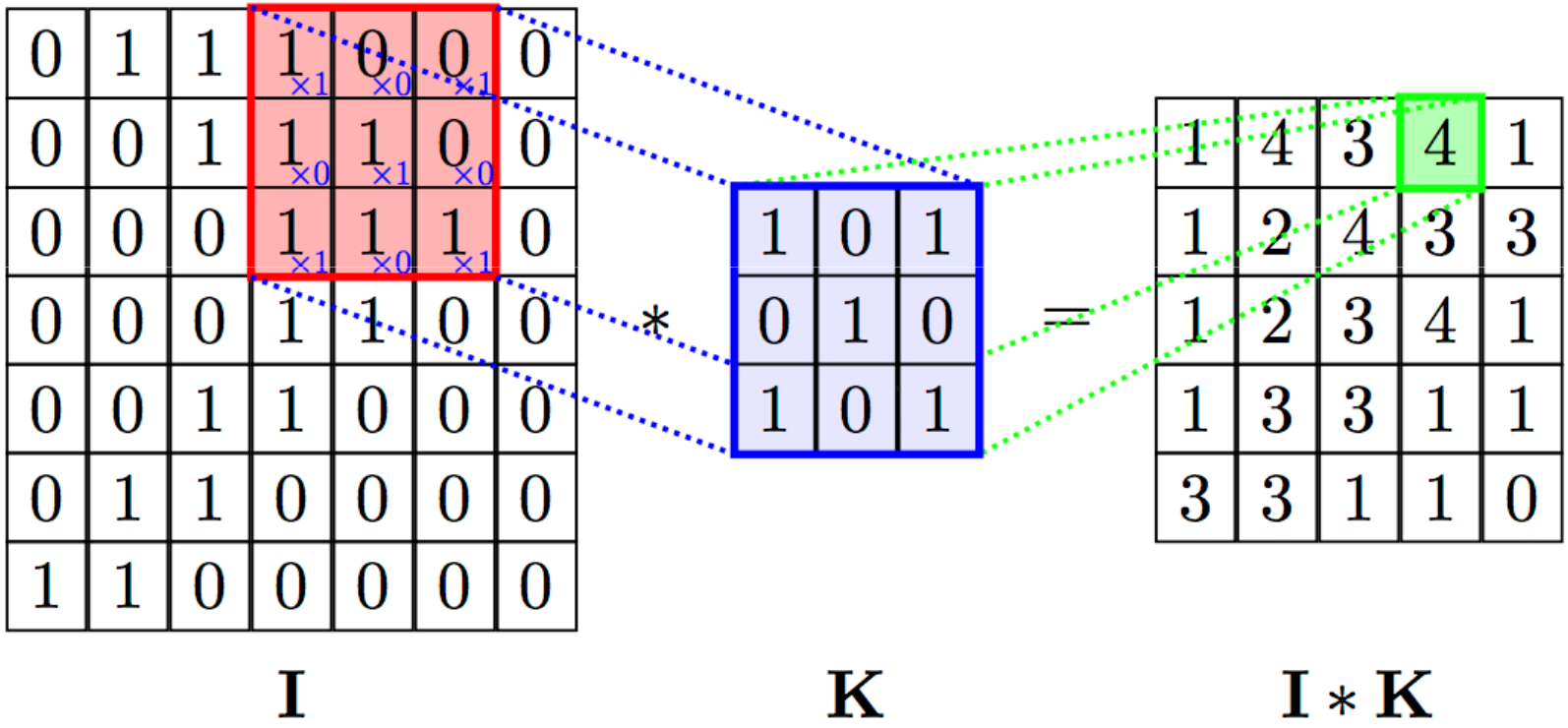


wyjście (y)



źródło: https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1

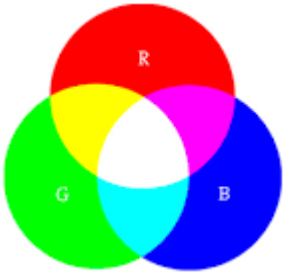
Splot 2D, wizualizacje



źródło: https://github.com/PetarV-/TikZ/tree/master/2D%20Convolution

Splot 2D, wizualizacje

	Input					Kernel			Intermediate Output			Output		
	1	0	1	0	2	0	1	0	7	5	3	19	13	15
	1	1	3	2	1	0	0	2	4	7	5	28	16	20
	1	1	0	1	1	0	1	0	7	2	8	23	18	25
	2	3	2	1	3									
	0	2	0	1	0									
	1	0	0	1	0	2	1	0	5	3	10			
	2	0	1	2	0	0	0	0	13	1	13			
	3	1	1	3	0	0	3	0	7	12	11			
	0	3	0	3	2									
	1	0	3	2	1									
	2	0	1	2	1	1	0	0	7	5	2			
	3	3	1	3	2	1	0	0	11	8	2			
	2	1	1	1	0	0	0	2	9	4	6			
	3	1	3	2	0									
	1	1	2	1	1									

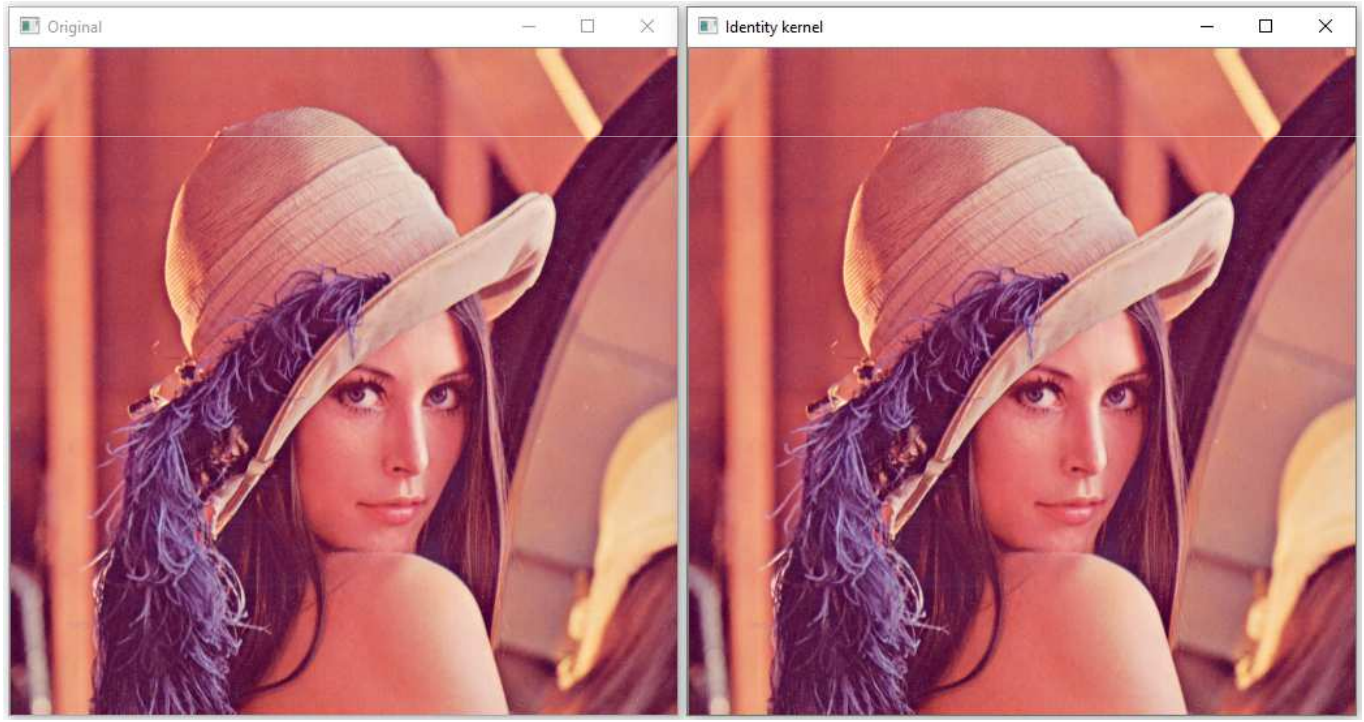


źródło: https://medium.com/apache-mxnet/multi-channel-convolutions-explained-with-ms-excel-9bbf8eb77108

Splot 2D, przetwarzanie obrazu

kernel neutralny

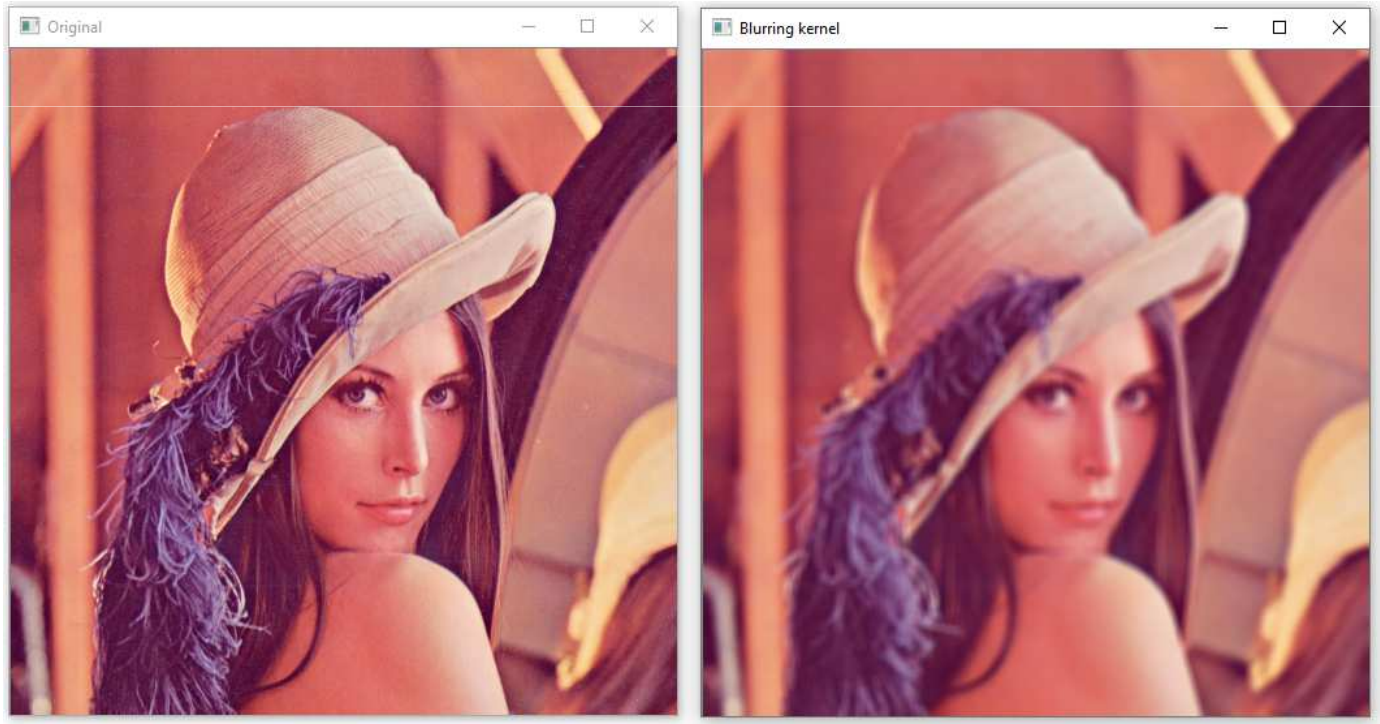
0	0	0
0	1	0
0	0	0



Splot 2D, przetwarzanie obrazu

kernel zmiękczający (Gaussian Blurring kernel)

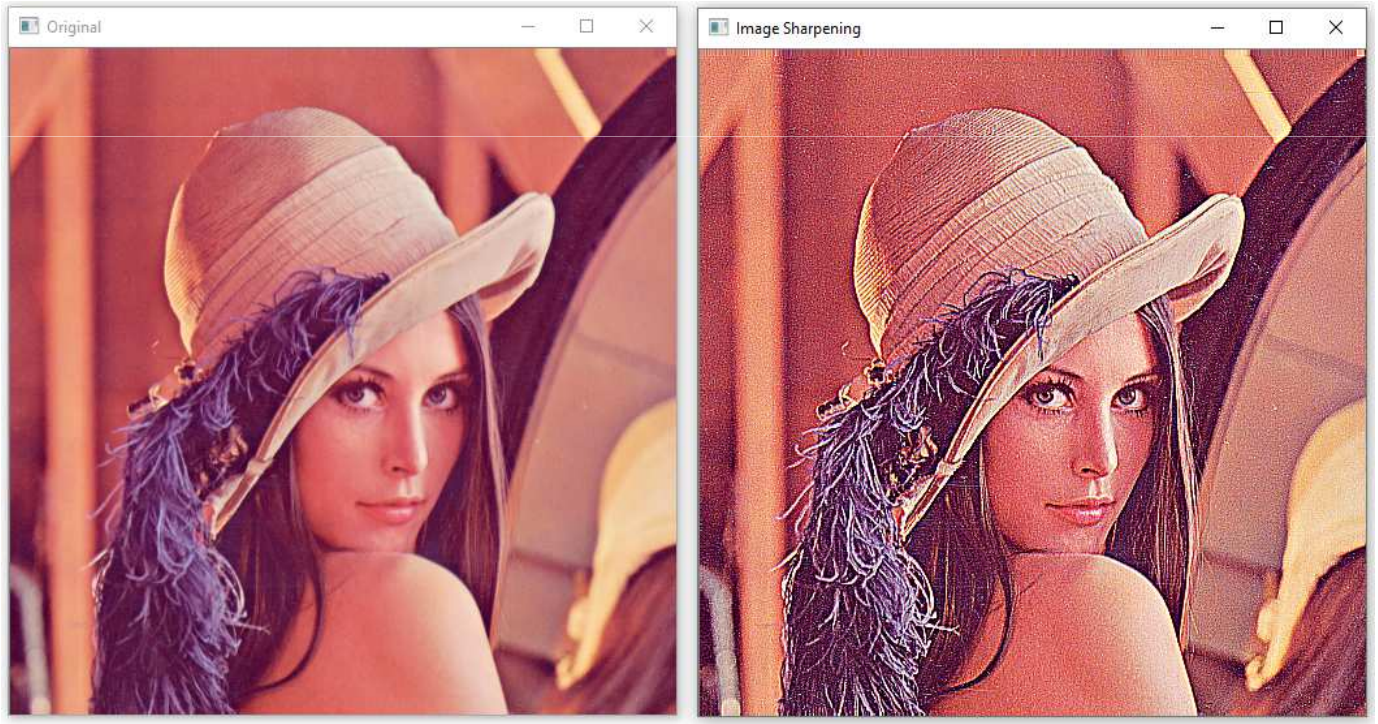
$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



Splot 2D, przetwarzanie obrazu

kernel wyostrzający (sharpen kernel)

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



Splot 2D, przetwarzanie obrazu

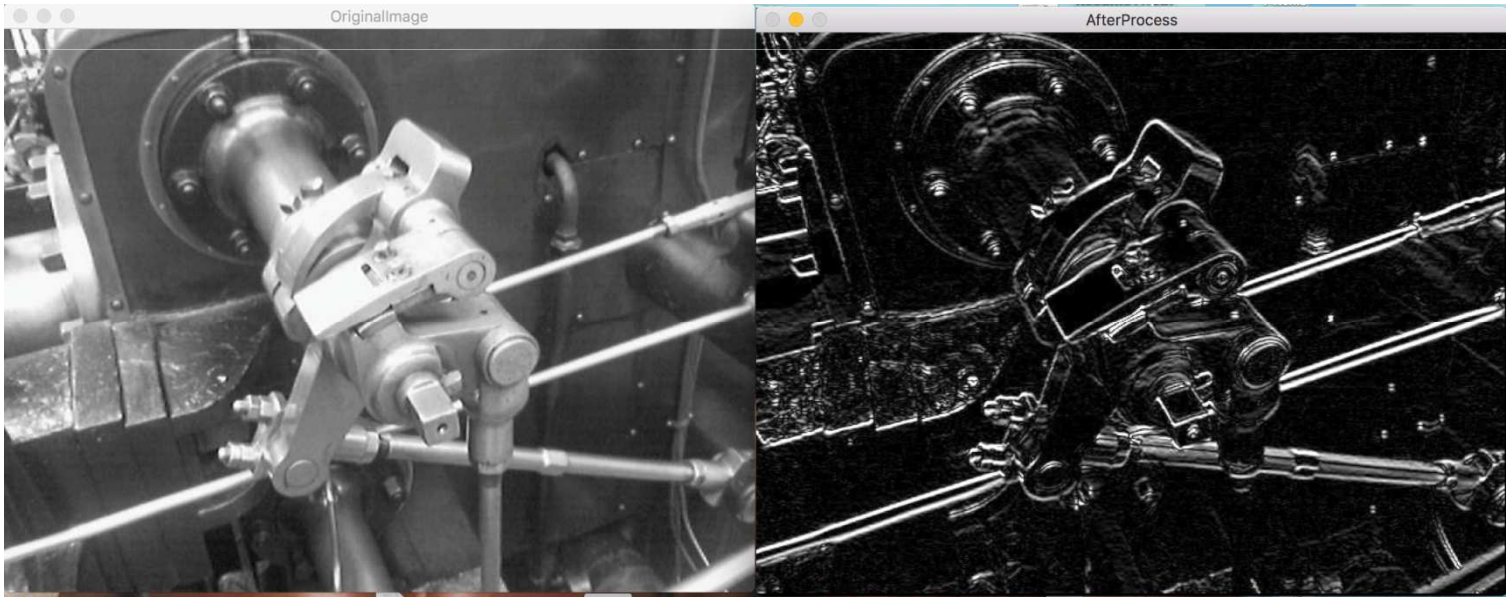
- Filtr Sobel-a (wykrywania krawędzi w cyfrowym przetwarzaniu obrazów)

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

x filter

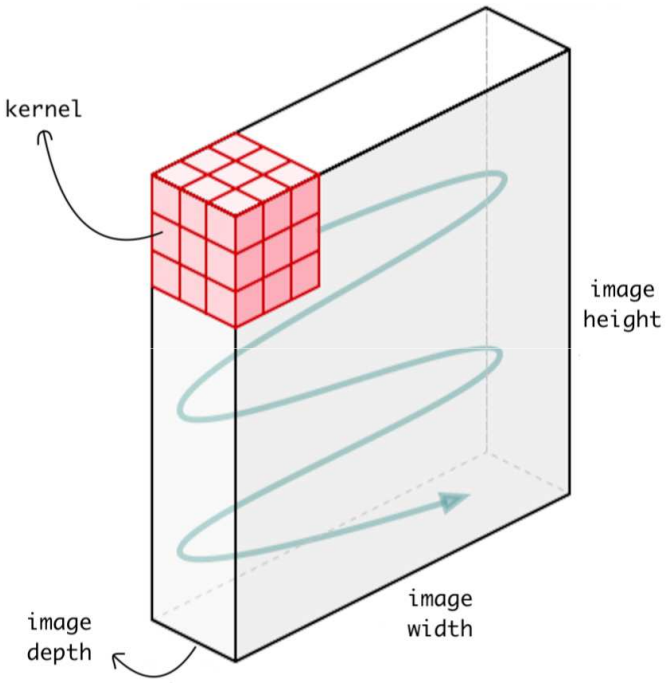
+1	+2	+1
0	0	0
-1	-2	-1

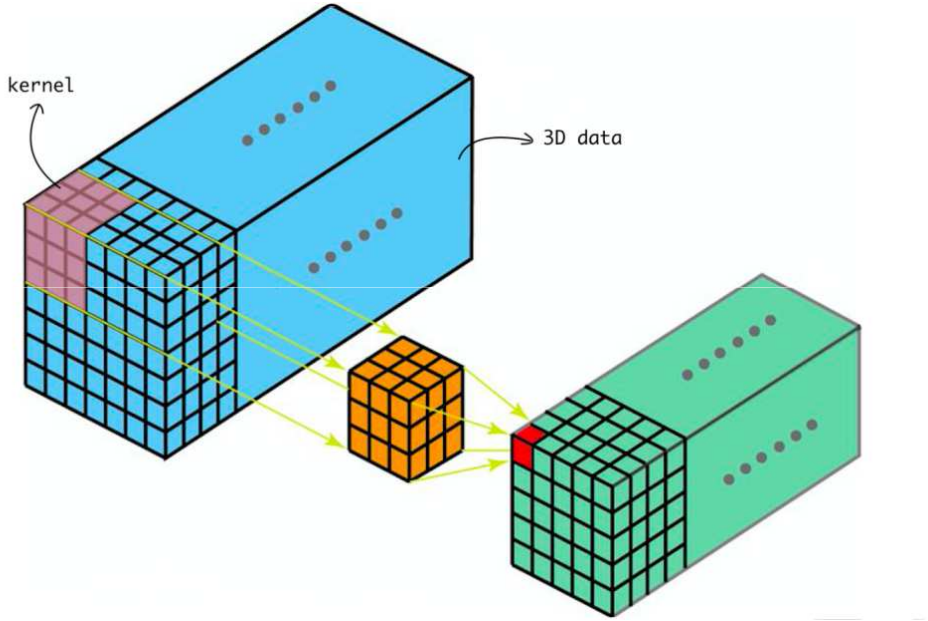
y filter



źródło: https://blog.saush.com/2011/04/20/edge-detection-with-the-sobel-operator-in-ruby/

Splot 3D, wizualizacje





źródło: https://mmuratarat.github.io/2019-01-17/implementing-padding-schemes-of-tensorflow-in-python