

Q.11 - 20

Q.11. 平滑化フィルタ

平滑化フィルタ(3x3)を実装せよ。

平滑化フィルタはフィルタ内の画素の平均値を出力するフィルタである。

入力 (imori.jpg)

出力 (answers_image/answer_11.jpg)



答え

- Python >> [answers_py/answer_11.py](#)
- C++ >> [answers_cpp/answer_11.cpp](#)

Q.12. モーションフィルタ

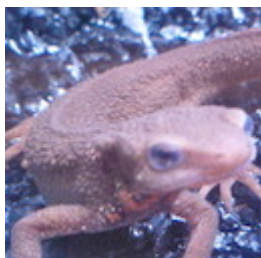
モーションフィルタ(3x3)を実装せよ。

モーションフィルタとは対角方向の平均値を取るフィルタであり、次式で定義される。

$$\begin{bmatrix} 1/3 & 0 & 0 \\ 0 & 1/3 & 0 \\ 0 & 0 & 1/3 \end{bmatrix}$$

入力 (imori.jpg)

出力 (answers_image/answer_12.jpg)



答え

- Python >> [answers_py/answer_12.py](#)
- C++ >> [answers_cpp/answer_12.cpp](#)

Q.13. MAX-MINフィルタ

MAX-MINフィルタ(3x3)を実装せよ。

MAX-MINフィルタとはフィルタ内の画素の最大値と最小値の差を出力するフィルタであり、**エッジ検出**のフィルタの一つである。エッジ検出とは画像内の線を検出することであり、このような画像内の情報を抜き出す操作を**特徴抽出**と呼ぶ。エッジ検出では多くの場合、グレースケール画像に対してフィルタリングを行う。

入力 (imori.jpg)

出力 (answers_image/answer_13.jpg)



答え

- Python >> [answers_py/answer_13.py](#)
- C++ >> [answers_cpp/answer_13.cpp](#)

Q.14. 微分フィルタ

微分フィルタ(3x3)を実装せよ。

微分フィルタは輝度の急激な変化が起こっている部分のエッジを取り出すフィルタであり、隣り合う画素同士の差を取る。

(a)縦方向

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

(b)横方向

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

入力 (imori.jpg)

出力・縦方向
(answers_image/answer_14_v.jpg)

出力・横方向
(answers_image/answer_14_h.jpg)



答え

- Python >> [answers_py/answer_14.py](#)
- C++ >> [answers_cpp/answer_14.cpp](#)

Q.15. Sobelフィルタ

Sobelフィルタ(3x3)を実装せよ。


ソーベルフィルタ(Sobelフィルタ)は特定方向（縦や横）のエッジのみを抽出するフィルタであり、次式でそれぞれ定義される。

(a)縦方向

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

(b)横方向

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

入力 (imori.jpg)	出力・縦方向 (answers_image/answer_15_v.jpg)	出力・横方向 (answers_image/answer_15_h.jpg)
		

答え

- Python >> [answers_py/answer_15.py](#)
- C++ >> [answers_cpp/answer_15.cpp](#)

Q.16. Prewittフィルタ

Prewittフィルタ(3x3)を実装せよ。


Prewittフィルタはエッジ抽出フィルタの一種であり、次式で定義される。

(a)縦方向

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

(b)横方向

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

入力 (imori.jpg)	出力・縦方向 (answers_image/answer_16_v.jpg)	出力・横方向 (answers_image/answer_16_h.jpg)
		

答え

- Python >> [answers_py/answer_16.py](#)
- C++ >> [answers_cpp/answer_16.cpp](#)

Q.17. Laplacian フィルタ

Laplacian フィルタを実装せよ。

Laplacian（ラプラシアン）フィルタとは輝度の二次微分をとることでエッジ検出を行うフィルタである。

デジタル画像は離散データであるので、x方向・y方向の一次微分は、それぞれ次式で表される。

$$\begin{aligned} I_x(x, y) &= (I(x+1, y) - I(x, y)) / ((x+1) - x) = I(x+1, y) - I(x, y) \\ I_y(x, y) &= (I(x, y+1) - I(x, y)) / ((y+1) - y) = I(x, y+1) - I(x, y) \end{aligned}$$

さらに二次微分は、次式で表される。

$$\begin{aligned} I_{xx}(x, y) &= (I_x(x, y) - I_x(x-1, y)) / ((x+1) - x) = I_x(x, y) - I_x(x-1, y) \\ &= (I(x+1, y) - I(x, y)) - (I(x, y) - I(x-1, y)) \\ &= I(x+1, y) - 2 * I(x, y) + I(x-1, y) \\ I_{yy}(x, y) &= \dots = I(x, y+1) - 2 * I(x, y) + I(x, y-1) \end{aligned}$$

これらより、ラプラシアン は次式で定義される。

$$\begin{aligned} D^2 I(x, y) &= I_{xx}(x, y) + I_{yy}(x, y) \\ &= I(x-1, y) + I(x, y-1) - 4 * I(x, y) + I(x+1, y) + I(x, y+1) \end{aligned}$$

これをカーネル化すると、次のようになる。

$$K = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

入力 (imori.jpg) 出力(answers_image/answer_17.jpg)



答え

- Python >> [answers_py/answer_17.py](#)
- C++ >> [answers_cpp/answer_17.cpp](#)

Q.18. Emboss フィルタ

Embossフィルタを実装せよ。

Embossフィルタとは輪郭部分を浮き出しにするフィルタで、次式で定義される。

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

入力 (imori.jpg)

出力(answers_image/answer_18.jpg)



答え

- Python >> [answers_py/answer_18.py](#)
- C++ >> [answers_cpp/answer_18.cpp](#)

Q.19. LoGフィルタ

LoGフィルタ(sigma=3、カーネルサイズ=5)を実装し、*imori_noise.jpg*のエッジを検出せよ。

LoGフィルタとはLaplacian of Gaussianであり、ガウシアンフィルタで画像を平滑化した後にラプラシアンフィルタで輪郭を取り出すフィルタである。

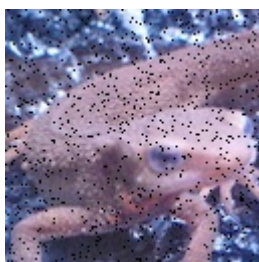
Laplacianフィルタは二次微分をとるのでノイズが強調されるのを防ぐために、予めGaussianフィルタでノイズを抑える。

LoGフィルタは次式で定義される。

$$\text{LoG}(x,y) = (x^2 + y^2 - \sigma^2) / (2 * \pi * \sigma^6) * \exp(-(x^2+y^2) / (2*\sigma^2))$$

入力 (imori_noise.jpg)

出力 (answers_image/answer_19.jpg)



答え

- Python >> [answers_py/answer_19.py](#)
- C++ >> [answers_cpp/answer_19.cpp](#)

Q.20. ヒストグラム表示

matplotlibを用いて`imori_dark.jpg`のヒストグラムを表示せよ。

ヒストグラムとは画素の出現回数をグラフにしたものである。matplotlibでは`hist()`という関数がすでにあるので、それを利用する。

入力 (imori_dark.jpg) 出力 (answers_image/answer_20.png)



答え >> [answers_py/answer_20.py](#)