Дата 2023-06-04

ФИО Клименко Иван Сергеевич

Группа 224-321

**ЛАБОРАТОРНАЯ РАБОТА №****4**  
**Применение пространственных фильтров размытия и повышения  
резкости**

**1. Цель работы**  
Познакомится с пространственными методами фильтрации.

**2. Исходные данные и программное обеспечение**

Изображения сгенерированы и обработаны программой на языке «Питон» с помощью библиотеки *NumPy и SciPy*.

**4. Выполнение работы**

[1, 2]. Периодический прямоугольный сигнал из значений {0.0, 1.0} замощается с помощью NumPy, в результате получается изображение 200×200 пк с глубиной цвета 32 бита (тип float32). Задан период 25 пк, одна полоса шириной 4 пк.

A picture containing text, screenshot, pattern, rectangle

Description automatically generated

Рисунок 1 – исходное изображение

3. В результате применения линейных сглаживающих фильтров, заданных функцией: f(*x, y*) = 1 / *xy*, получились изображения на рис. 2 для разных значений *x, y*.

A picture containing screenshot

Description automatically generated

Рисунок 2 – сглаженное изображения при параметрах:  
(2, 2); (5, 5);  
(16, 16); (100, 100).

4. Наложение импульсного шума реализовано в две операции:

1) формирование битовой маски соответствующий размеру изображения со случайным нормальным распределением, с P(1) = 20%;  
 2) наложение маски на изображение с помощью операции XOR.

A picture containing text, pattern, screenshot, black and white

Description automatically generated

Рис. 3 – изображение, модулированное импульсным шумом

5. Учитывая монотонность изображения по горизонтали, можно задать такие параметры ядра медианного фильтра, при которых будет полностью восстановлен исходный образ.

A picture containing text, pattern, black and white, screenshot

Description automatically generated

Рис. 4 – зашумлённое изображение после медианной фильтрации, ядра:  
5×5; 20×1;  
20×20; 40×1.

6. Размытые изображения для повышения резкости обработаны фильтром Лапласа.

A picture containing text, screenshot

Description automatically generated

Рис. 5 – изображения из рис. 2 после сглаживания и обработки фильтром Лапласа.

**Исходный код**

import numpy as np

import scipy as sp

import matplotlib.pyplot as plt

# вспомогательные функции

emap = lambda f, xs: [f(x) for x in xs]

unzip = lambda xs: [[a for a, \_ in xs], [b for \_, b in xs]]

uint8 = lambda x: x.astype(np.uint8)

float32 = lambda x: x.astype(np.float32)

show1 = lambda pic: (plt.subplots())[1].imshow(pic, cmap='gray')

def show(pics, title=None):

cols, rows = 2, (len(pics) // 2) + (len(pics) % 2)

fig, axs = plt.subplots(rows, cols, squeeze=True, constrained\_layout=True)

show1 = lambda pic, ax: ax.imshow(pic, cmap='gray')

[show1(pic, ax) for pic, ax in zip(pics, axs.flat)]

def save(pics, suffix='', path='./', ext='gif'):

for n, p in enumerate(pics):

imsave(path + f'{n+1:02d}{suffix}.{ext}', p)

# создание прямоугольного сигнала

def square\_wave(period, w=200, h=200):

freq = w // period

tile = np.repeat((0.0, 1.0), freq // 2)

line = np.tile(tile, period)

pic = np.repeat(line, h).reshape(w, h).T

print(f"Период: {period} пк, полоса: {freq // 2} пк")

return pic

pic = square\_wave(25)

show1(pic)

# сглаживание

def blur(pic, x, y):

f = lambda x, y: np.ones((x, y)) / (x \* y)

return sp.signal.convolve2d(pic, f(x, y))

blurred = [blur(pic, x, x) for x in (2, 5, 16, 100)]

show(blurred)

# зашумление

def noise(pic, p):

mask = np.random.choice((0, 255), size=pic.shape, p=(1-p, p))

noised = uint8(pic) \* 255 ^ mask

return float32(noised // 255)

noisy = noise(pic, .20)

show1(noisy)

# медианная фильтрация

mf = lambda a, b: sp.ndimage.median\_filter(noisy, size=(a, b))

mf = [mf(a, b) for a, b in ((5, 5), (20, 1), (20, 20), (40, 1))]

show(mf)

# фильтр Лапласа

laplaced = [sp.ndimage.laplace(p) for p in blurred]

show(laplaced)