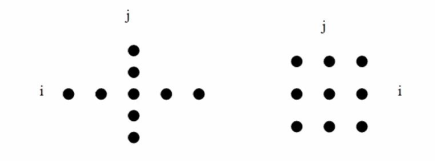
**Обробка зображень**

**Лабораторна робота №2**

**Грищенко Юрій, ІПС-32**

Шум імпульсного характеру можна досить легко та ефективно подавити за допомогою медіанної фільтрації:



Відбувається послідовна обробка кожної точки зображення, розглядається двовимірне вікно навколо точки. Часто використовується вікно у вигляді хреста або квадрату.

Знаходимо медіану множини точок, що належать цьому вікну — ця медіана і буде продуктом фільтрації.

Якщо імпульсний шум покриває область розміром NxM, то його буде подавлено при розмірі вікна 2Nx2M.

Отже, для подавлення шуму у вигляді точок 2x2 можна використати звичайний квадрат 5x5 (код на Python):

import numpy as np

from PIL import Image

import sys

if len(sys.argv) <= 3:

print("Usage: lab2.py [input] [output] [filterRadius]")

exit(1)

filterRadius = int(sys.argv[3])

image = Image.open(sys.argv[1]).convert("L")

inputArr = np.asarray(image)

outputArr = np.empty\_like(inputArr)

for y in range(inputArr.shape[1]):

for x in range(inputArr.shape[0]):

inputSlice = inputArr[max(0, x - filterRadius) :

min(inputArr.shape[0] - 1, x + filterRadius),

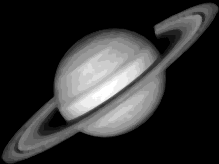
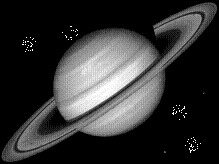
max(0, y-filterRadius) :

min(inputArr.shape[1] - 1, y + filterRadius)]

outputArr[x, y] = np.median(inputSlice)

Image.fromarray(outputArr, "L").save(sys.argv[2])

Зображення до та після фільтрації: **радіус 2**



Для шуму, що покриває велику область, доцільніше використати не великий квадрат, а великий хрест (таким чином алгоритм потребуватиме менше операцій):

import numpy as np

from PIL import Image

import sys

if len(sys.argv) <= 3:

print("Usage: lab2.py [input] [output] [filterRadius]")

exit(1)

# Configurable radius

filterRadius = int(sys.argv[3])

image = Image.open(sys.argv[1]).convert("L")

inputArr = np.asarray(image)

outputArr = np.empty\_like(inputArr)

for y in range(inputArr.shape[1]):

for x in range(inputArr.shape[0]):

aperture = []

for i in range(max(0, x-filterRadius), min(inputArr.shape[0], x+filterRadius+1)):

aperture.append(inputArr[i, y])

for i in range(max(0, y-filterRadius), min(inputArr.shape[1], y+filterRadius+1)):

if i != y:

aperture.append(inputArr[x, i])

outputArr[x, y] = np.median(np.array(aperture))

Image.fromarray(outputArr, "L").save(sys.argv[2])

Зображення до та після фільтрації: **радіус 3**

