**ЛАБОРАТОРНА РОБОТА № 6**

**СИНТЕЗУВАННЯ ЗОБРАЖЕНЬ**

**Мета роботи** – набуття навичок синтезування цифрових зображень, генерації шуму різної природи виникнення та використання базових арифметичних та геометричних операцій над цифровим зображенням.

## **6.1 Короткі теоретичні відомості**

### **6.1.1 Синтезування зображень**

Під синтезованим цифровим зображенням розуміється будь-яке візуальне подання інформації, одержуване в результаті обчислень в певній обчислювальній системі (комп'ютері), яка виконує деякі директиви користувача (одержувача зображення), поза фізичним зв'язком з реальним зображенням. Останнє зауваження виключає з предмета цифрової обробки зображень такі завдання, як обробка зображень, одержуваних при фотографії або відеозйомки в реальному часі, оскільки в даному випадку мають справу з фізичними об'єктами, а не з побудовою образів графічних об'єктів за тими чи іншими описами. Наприклад, описом кола, зображення якого треба отримати на екрані монітора, може бути його рівняння, або координати центру і радіус, або двійкове подання її контуру.

Безліч видимих на екрані елементів, упорядкованих згідно з певними просторовими відносинами (як такі відносини можна, наприклад, назвати відносини «далі -ближче», «більше-менше» і т.д.), із заданими атрибутами оточення (наприклад, рівень освітленості, ознака наявності серпанку та її щільності), з атрибутами поверхонь елементів (колір, текстура, оптичні властивості матеріалів об'єктів, що моделюються) утворюють сцену. Графічні об'єкти, з яких складається сцена, будемо називати елементами сцени.

Таким чином, *синтезування зображення* – це процес створення візуального образу шляхом програмного заповнення матриці (що є зображенням) числовими значеннями. Напівтонове зображення представляється матрицею, де 0 – це чорний колір, а 1 (або 255) – білий. В Scilab реалізується за допомогою функцій:

* **zeros(M, N)** – створює чорне зображення розміром M × N;
* **ones(M, N)** – створює біле зображення розміром M – N.

*Пряме присвоєння* використовується для створення об'єктів (наприклад, білий квадрат):

**зображення(рядки, стовпці) = 1.**

### **6.1.2 Цифровий шум зображень**

Однією з головних складових зображення є шум. Шумом називається дефект зображення, внесений впливом навколишнього середовища, фотосенсорами та електронікою пристроїв, що їх використовують, внаслідок недосконалості технологій. Шум проявляється на зображенні у вигляді накладеної маски з пікселів випадкового кольору та яскравості. Синтезування шуму на зображенні прийнято реалізовувати на основі заданої функції ймовірності густини або функції розподілу. Більшість генераторів випадкових чисел у мовах програмування (включаючи SciLab) засновані на переформулюванні завдання у термінах випадкових чисел з рівномірною функцією розподілу в інтервалі [0, 1] – за допомогою функції **rand**. На основі рівномірного закону розподілу можуть бути отримані будь-які види розподілу шумів. У основі цього підходу лежить одне з тверджень теорії ймовірності о тому, що якщо є випадкова величина з рівномірним розподілом на відрізку   
[0, 1], то випадкову величину *z* з заданою функцією розподілу можна побудувати за формулою:

,

тобто. в еквівалентної формі необхідно вирішити рівняння виду щодо *z*.

Наприклад, для генерації випадкової величини *z* з функцією розподілу ймовірностей Релея, слід скористатися такою формулою:



При використання випадкових чисел з рівномірним розподілом в інтервалі [0, 1], щоб отримати *z*, достатньо розв'язати рівняння:

 або 

Оскільки квадратний корінь – позитивна функція, то генеруються випадкові величини будуть завжди більше *a*, що потрібно у визначенні функції Релея. Отже, рівномірно розподілені випадкові числа зможуть служити основою для генератора релеєвських випадкових чисел з заданими параметрами *a* і *b*.

Вираз виду  іноді називають рівнянням генератора випадкових чисел, оскільки у ньому визначається, як обчислювати необхідні випадкові величини. У цьому випадку є проста формула для рішення рівняння. Однак це не завжди можливо, і проблему можна сформулювати наступним чином: як отримати рівняння генератора випадкових чисел, вихід якого добре наближає випадкову величину із заданою функцією щільності ймовірностей.

У таблиці наведено форми уявлення випадкової величини. Для деяких з них вдається виписати формулу зворотної функції розподілу, наприклад, для експоненційного розподілу або функції розподілу Релея. У цих випадках є проста формула для вираження необхідних випадкових чисел термінах рівномірних випадкових чисел. В інших випадках, як у випадку гаусової чи логарифмічно нормальної густини, такої простої формули не існує. Тоді необхідно шукати альтернативний спосіб – наприклад, для випадковий величини *z* з логарифмічно нормальною щільністю можна скористатися тим фактом, що величина  має гаусово розподіл, і виписати вираз в термінах гаусовий випадково величини з нульовим середнім та одиничною дисперсією.

Позначують, що  – нормальна (Гаусова) випадкова величина зі середнім нуль і дисперсією 1, а  – рівномірна випадкова величина з інтервалу (0, 1).

Представлені далі генератори випадкової величини відіграють важливу роль при моделюванні поведінки випадкового шуму у програмах обробки зображень. Так гаусів шум використовується як природне наближення в тих випадках, коли детектори зображення працюють на порозі чутливості. Шум типу «сіль та перець» виникає в пристроях з помилковою комутацією. Розміри зерен на фотоемульсії є випадковими величинами з логарифмічно нормальним законом розподілу. Шум Релея утворюється при фіксації віддалених зображень, а експоненційний шум і шум Ерланга використовуються при описі спотворень на зображеннях, отриманих лазерним випромінюванням.

Для моделювання шуму використовують різні закони розподілу ймовірностей (Probability Density Function, PDF – рис. 6.1).

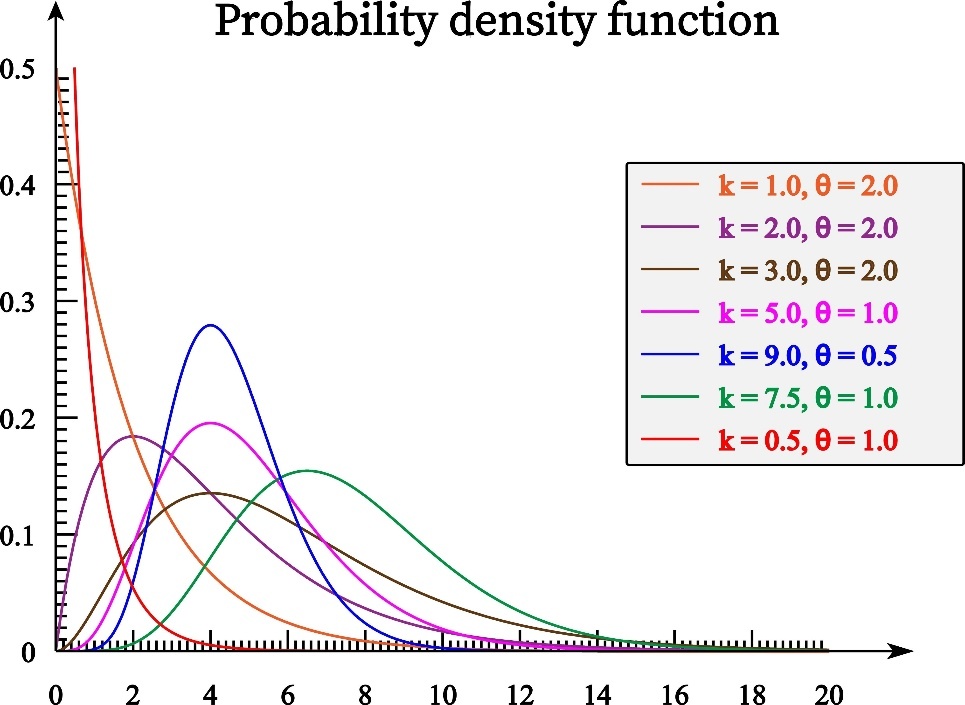


Рисунок 6.1 – Графчне відображення розподілу ймовірностей за різними законами

#### 6.1.2.1 Рівномірна випадкова величина

Рівномірна випадкова величина можне бути описана формулою:



Середнє і дисперсія для цієї випадкової величини обраховуються як:



Розподіл:



В SciLab генерація рівномірної випадкової величини може бути виконана функцією **rand(M, N).**

#### 6.1.2.2 Гаусіва випадкова величина

Гаусіва випадкова величина можне бути описана формулою:



Середнє і дисперсія для цієї випадкової величини обраховуються як:



Розподіл:



В SciLab генерація гаусівої випадкової величина може бути виконана функцією **grand(M, N, 'nor', mean, std\_dev).**

#### 6.1.2.3 Випадкова величина «сіль та перець»

Випадкова величина «сіль та перець» можне бути описана формулою:



Середнє і дисперсія для цієї випадкової величини обраховуються як:



Розподіл:



В SciLab генерація випадкової величини «сіль та перець» може бути виконана функцією **rand(M, N, 'uniform').**

#### 6.1.2.4 Логарифмічно нормальний випадкова величина

Логарифмічна випадкова величина можне бути описана формулою:



Середнє і дисперсія для цієї випадкової величини обраховуються як:



Розподіл:



Генерація логарифмічної випадкової величини може бути виконана функцією:



Команда SciLab:

**R=exp(grand([M,N],’nor’,μ,σ))**

де:

μ – середнє значення нормального розподілу (логарифма R);

σ – стандартне відхилення нормального розподілу (логарифма R).

#### 6.1.2.5 Випадкова величина Релея

Випадкова величина Релея можне бути описана формулою:



Середнє і дисперсія для цієї випадкової величини обраховуються як:



Розподіл:



Генерація випадкової величини Релея може бути виконана функцією:



Команда SciLab:

**R = a + (-b \* log(1 - rand(M, N, 'uniform'))).^0.5;**

#### 6.1.2.6 Експоненціальна випадкова величина

Експоненціальна випадкова величина можне бути описана формулою:



Середнє і дисперсія для цієї випадкової величини обраховуються як:



Розподіл:



Генерація експоненціальної випадкової величини може бути виконана функцією:



На відміну від Гаусівського шуму, який симетричний навколо середнього, експоненціальний шум є одностороннім і має довгий "хвіст" у бік великих позитивних значень, що часто виражається у вигляді світлих плям на темному тлі.

Команда SciLab:

**k = -1/a; // Де k — коефіцієнт масштабування**

**R = k \* log(1 - rand(M, N, 'uniform'));**

#### 6.1.2.7 Випадкова величина Ерланга

Випадкова величина Ерланга можне бути описана формулою:



Середнє і дисперсія для цієї випадкової величини обраховуються як:



Розподіл:



Генерація випадкової величини Ерланга може бути виконана функцією:



де  – експоненційні величини з параметром a.

Команда SciLab:

**k = -1/a;**

**R = zeros(M, N);**

**for j = 1:b**

**R = R + k \* log(1 - rand(M, N, 'uniform'));**

**end**

#### 6.1.2.8 Приклад коду функції для генерації різних шумів на зображенні

Лістинг функції генерації шумів

function **R**=imnoise\_scilab\_exact(**type**, **M**, **N**, **a**, **b**)

// IMNOISE2 Генерує масив випадкових чисел із заданою

// Probability Density Function ( Функція густини ймовірності ).

// R = imnoise\_scilab\_exact(TYPE, M, N, A, B)

// Цей код є перекладом функції IMNOISE, описаної в книзі:

// R. C. Gonzalez, R. E. Woods, & S. L. Eddins, "Digital Image

// Processing Using MATLAB," Prentice-Hall, 2004.

num\_args = argn(2);

// Ініціалізація R нулями на випадок помилки

**R** = zeros(**M**, **N**);

// Встановлення значень за замовчуванням.

if num\_args == 1 then

**a** = 0; **b** = 1;

**M** = 1; **N** = 1;

else

if num\_args == 3 then

**a** = 0; **b** = 1;

end

end

// Початок обробки. Використовуємо convstr(type, 'l') для захисту від вводу з великої

// літери.

select convstr(type, 'l')

case 'uniform'

**R** = **a** + (**b** - **a**)\*rand(**M**, **N**, 'uniform');

case 'gaussian'

// Використовуємо перетворення Бокса-Мюллера через нестабільність grand

// Генеруємо випадкові числа парами (для Бокса-Мюллера)

U1 = rand(**M**, **N**, 'uniform');

U2 = rand(**M**, **N**, 'uniform');

// Гаусівський шум з середнім 0 і ст. відх. 1

Z0 = sqrt(-2.0 \* log(U1)) .\* cos(2.0 \* %pi \* U2);

// Масштабуємо та зміщуємо

**R** = **a** + **b** \* Z0;

case 'salt & pepper'

if num\_args <= 3 then

**a** = 0.05; **b** = 0.05;

end

// Перевірка, що Pa + Pb не перевищує 1.

if (**a** + **b**) > 1 then

error('Сума Pa + Pb не повинна перевищувати 1.')

end

// Ініціалізація, яка гарантує, що R не буде пустою

**R**(1:**M**, 1:**N**) = 0.5;

// Генеруємо рівномірно розподілені випадкові числа

X = rand(**M**, **N**, 'uniform');

// Перець (значення 0)

c = find(X <= **a**);

**R**(c) = 0;

// Сіль (значення 1)

u = **a** + **b**;

c = find(X > **a** & X <= u);

**R**(c) = 1;

case 'lognormal'

if num\_args <=3 then

**a** = 1; **b** = 0.25;

end

// Аналогічно, використовуємо Бокса-Мюллера для генерації нормального розподілу

U1 = rand(**M**, **N**, 'uniform');

U2 = rand(**M**, **N**, 'uniform');

Z0 = sqrt(-2.0 \* log(U1)) .\* cos(2.0 \* %pi \* U2);

// Логнормальний розподіл

**R** = exp(**b** \* Z0 + **a**);

case 'rayleigh'

**R** = **a** + (-**b**\*log(1 - rand(**M**, **N**, 'uniform'))).^0.5;

case 'exponential'

if num\_args <= 3 then **a** = 1;

end

if **a** <= 0 then

error('Параметр a повинен бути позитивним для експоненційного типу.')

end

k = -1/**a**;

**R** = k\*log(1 - rand(**M**, **N**, 'uniform'));

case 'erlang'

if num\_args <= 3 then

**a** = 2; **b** = 5;

end

if (**b** ~= round(**b**) | **b** <= 0) then

error('Параметр b має бути додатним цілим числом для Ерланга.')

end

k = -1/**a**;

**R** = zeros(**M**, **N**);

for j = 1:**b**

**R** = **R** + k\*log(1 - rand(**M**, **N**, 'uniform'));

end

else // otherwise

error('Невідомий тип розподілу.')

end // end select

endfunction

Лістинг із прикладом використанні функції генерації шумів

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// ВАЖЛИВО: Перед виконанням цього файлу (lab6.sci)

// ПЕРЕКОНАЙТЕСЯ, ЩО ВИ ЗАВАНТАЖИЛИ ФУНКЦІЮ, виконавши:

// exec('D:\temp\_SciLab\imnoise\_scilab\_exact.sci', -1)

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

imnoise\_func = 'imnoise\_scilab\_exact'; // Явно вказуємо назву функції

mprintf('Виконання прикладів з використанням функції: %s\n', imnoise\_func);

// ----------------------------------------------------

// Приклад 1: Гаусів шум

// ----------------------------------------------------

M = 100;

N = 100;

// Формуємо команду за допомогою sprintf, щоб уникнути помилок конкатенації

command\_gaussian = sprintf('Noise\_Gaussian = %s(''%s'', %d, %d, 0.1, 0.05);', imnoise\_func, 'gaussian', M, N);

execstr(command\_gaussian);

mprintf('\nПерші 5x5 елементів матриці Гаусового шуму:\n');

disp(Noise\_Gaussian(1:5, 1:5));

// Візуалізація гістограми

figure(1);

histplot(50, Noise\_Gaussian(:));

title('Гістограма Гаусового шуму (Середнє 0.1, Ст. відх. 0.05)');

xgrid();

// ----------------------------------------------------

// Приклад 2: Шум Сіль та Перець

// ----------------------------------------------------

M = 5;

N = 5;

command\_sp = sprintf('Noise\_SaltAndPepper = %s(''%s'', %d, %d);', imnoise\_func, 'salt & pepper', M, N);

execstr(command\_sp);

mprintf('\nМатриця шуму Сіль та Перець (5x5):\n');

disp(Noise\_SaltAndPepper);

// Візуалізація гістограми

figure(1);

histplot(50,Noise\_SaltAndPepper(:));

title('Гістограма шуму Сіль та Перець');

xgrid();

// ----------------------------------------------------

// Приклад 3: Рівномірний Шум

// ----------------------------------------------------

M = 2;

N = 6;

command\_uniform = sprintf('Noise\_Uniform = %s(''%s'', %d, %d, -0.2, 0.2);', imnoise\_func, 'uniform', M, N);

execstr(command\_uniform);

mprintf('\nМатриця рівномірного шуму (2x6):\n');

disp(Noise\_Uniform);

// Візуалізація гістограми

figure(1);

histplot(50, Noise\_Uniform (:));

title('Гістограма рівномірного шуму ');

xgrid();

Результати виконання:

--> exec('D:\temp\_SciLab\imnoise\_scilab\_exact.sci', -1)

--> exec('D:\temp\_SciLab\lab6.sci', -1)

Виконання прикладів з використанням функції : imnoise\_scilab\_exact

Перші 5x5 елементів матриці Гаусового шуму:

0.1399061 0.1463993 0.029868 0.2765284 0.128579

0.1206534 0.1449573 0.0915937 0.1328342 0.0657003

0.0215577 0.0681493 0.1013846 0.1659196 0.2214462

0.1453477 0.1890468 0.1336433 0.1294264 0.1191848

0.0585679 0.0771233 0.1274974 0.1215475 0.0920688

Матриця шуму Сіль та Перець (5x5):

0. 0.5 0.5 0.5 0.5

0.5 0.5 0. 0. 0.5

0.5 0.5 0.5 0.5 0.5

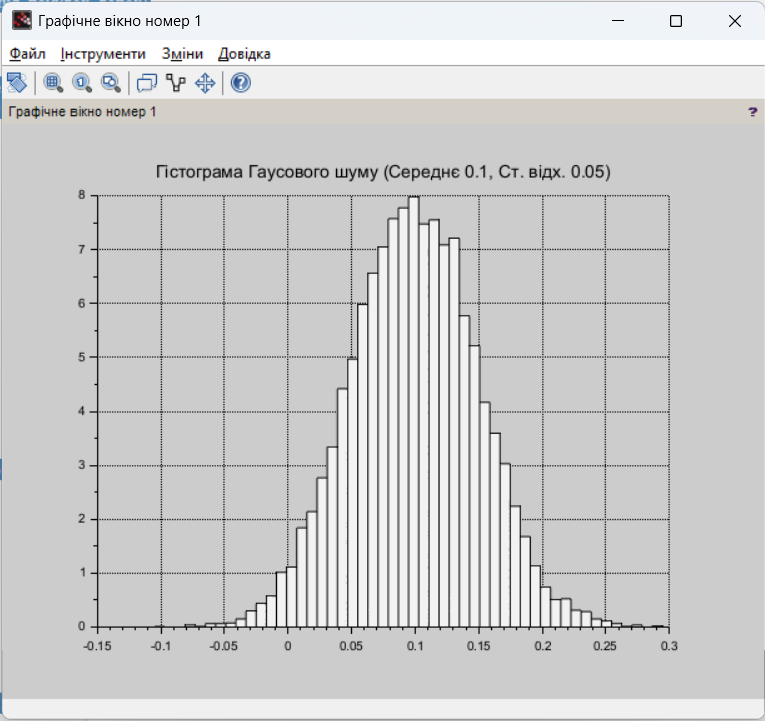
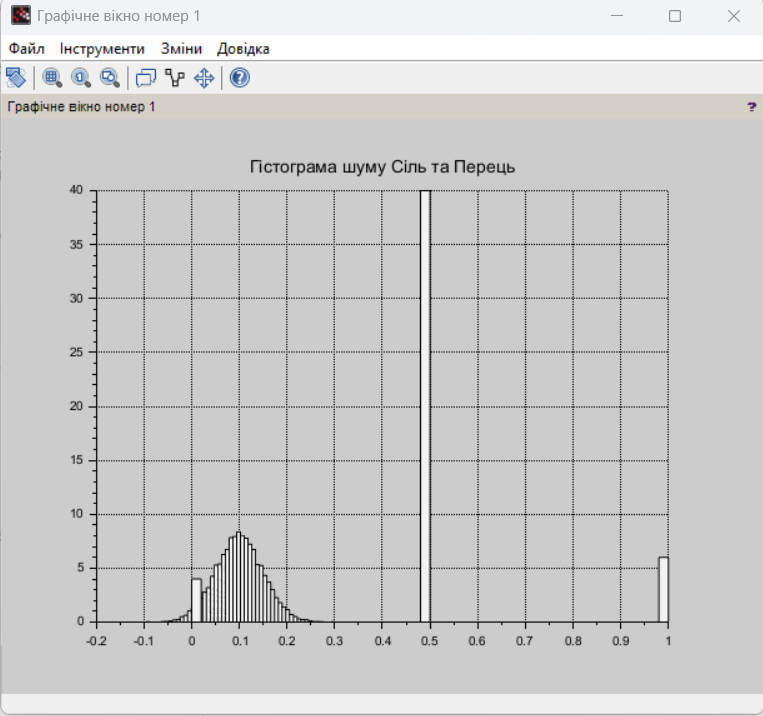
0.5 0.5 0.5 0.5 0.5

0. 0.5 0.5 1. 0.5

Матриця рівномірного шуму (2x6):

0.0996382 -0.1290816 -0.1617289 -0.0376124 0.1122252 0.0396324

0.1566084 -0.1319128 0.0538885 -0.0379043 0.1610549 -0.0932685

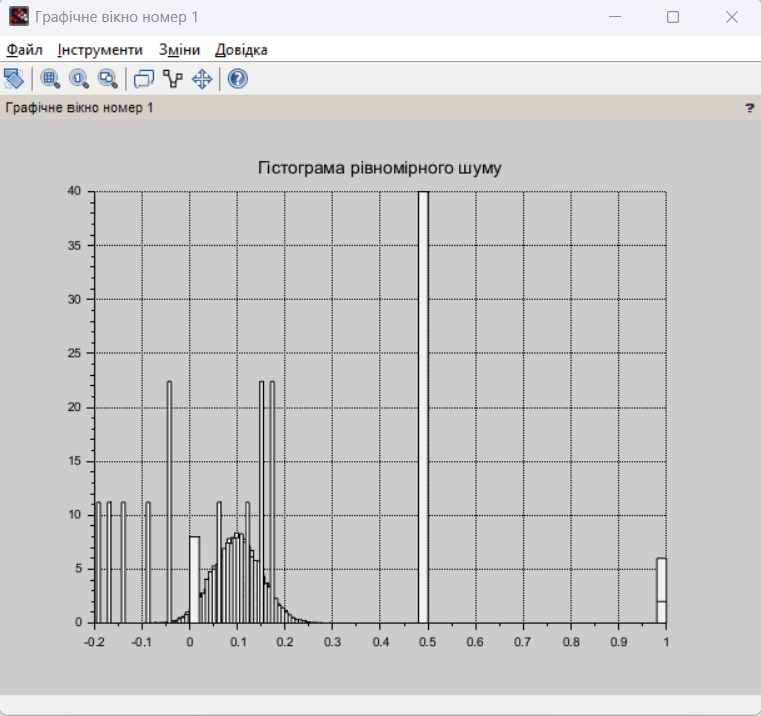


Рисунок 6.1 – Графічне відображення шумів на графіках

### **6.1.3 Гістограма зображення**

*Гістограма* – це графічне представлення розподілу рівнів інтенсивності (яскравості) пікселів у цифровому зображенні.

Вісь X– це діапазон значень яскравості (наприклад, 0 до 255).

Вісь Y– це кількість пікселів, що мають відповідне значення яскравості.

Аналіз гістограми дозволяє оцінити контрастність і загальну яскравість зображення. Наприклад, скупчення пікселів у лівій частині свідчить про переважання темних тонів.

Для побудови у Scilab гістограми використовується функція:

**// 1. Отримання даних гістограми (count - кількість пікселів, x\_values - значення яскравості)**

**[count, x\_values] = imhist(img);**

**// 2. Побудова графіка**

**plot(x\_values, count);**

*Геометричні перетворення* змінюють просторове розташування пікселів зображення.

*Афінні перетворення* – це узагальнений клас перетворень, які зберігають прямі лінії та паралельність, але можуть змінювати довжину та кути. Вони описують такі операції, як обертання, масштабування, зсув (translation) та зсув (shear).

Перетворення точки (*x*, *y*) у нову точку (*x*', *y*') задається матричною формою:

.

Приклади афінних перетворень:

1. *відображення* (Reflection / Flip):

* *горизонтальне відображення*;

Для реалізації у Scilab використовується індексування з кроком -1 для стовпців.

**img\_horizontal = img(:, end:-1:1);**

* *вертикальне відображення.*

Реалізація у Scilab: Використовується індексування з кроком -1 для рядків.

**img\_vertical = img(end:-1:1, :);**

1. *поворот* на 90° (Rotation).

Поворот виконується через транспонування та відображення. Наприклад, поворот на 90° за годинниковою стрілкою (CW) в Scilab можна виконати за допомогою наступної команди:

**img\_90cw = img'(:, end:-1:1);**

*Арифметичні операції* виконуються попіксельно над одним або двома зображеннями. Це такі операції як:

1. «*Негатив*» (Negative) – інвертує рівні яскравості, для 8-бітового зображення:



Реалізація у Scilab:

**img\_negative = 255 - img;**

1. *Масштабування яскравості* (Brightness Scaling) – зміна загальної яскравості шляхом множення або ділення на константу C. Наприклад, *ділення* (зменшення яскравості) у Scilab реалізується так:

**img\_dimmed = img / C; // Наприклад, C=4**

1. *Операція* *різниці* (Difference) – попіксельне віднімання, яке виділяє відмінності між двома зображеннями *I*1 та *I*2:



Реалізація у Scilab:

**img\_diff = abs(img1 - img2);**

## **6.2 Практична робота**

1. Створити необхідні папки для збереження вихідних зображень.
2. Синтезувати порожнє напівтонове 8-бітове зображення заданого розміру (наприклад, 800x800).
3. Додати на порожнє зображення шум згідно з індивідуальним завданням.
4. Побудувати гістограму розподілу отриманого зображення шуму.
5. Синтезувати зображення Об'єкта 1 у центрі зображення (білого кольору).
6. Синтезувати зображення Об'єкта 2 у верхньому лівому куті.
7. Синтезувати зображення Об'єкта 3 у правому нижньому куті.
8. Виконати операцію горизонтального відображення синтезованого зображення.
9. Виконати операцію вертикального відображення синтезованого зображення.
10. Виконати операцію повороту синтезованого зображення на 90 градусів за годинниковою стрілкою.
11. Виконати операцію повороту синтезованого зображення на 90 градусів проти годинникової стрілки.
12. Створити або вирізати з реального фону (або створити штучний фон) ділянку розміром 800x800.
13. Зменшити яскравість вилученого фону в 4 рази.
14. Сформувати нове напівтонове зображення з вилученого фону із зменшеною яскравістю та додати на нього шум. Зберегти.
15. Виконати операцію «Негатив» над зображенням, отриманим у п. 14. Зберегти.
16. Сформувати нове напівтонове зображення з вилученого фону із зменшеною яскравістю, додати на нього один із синтезованих об'єктів та шум. Зберегти.
17. Виконати операцію різниці двох зображень, отриманих у пп. 14 та 16. Зберегти результат.

Рисунок 6.2 – Розташування синтезованих об'єктів на зображенні

*Примітка.* Синтезований об'єкт має бути білого кольору. Розміри об'єкта вибираються таким чином, щоб він був помітний на зображенні.

Таблиця 6.1 – Варіанти індивідуального завдання

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Об'єкт 1 | Об'єкт 2 | Об'єкт 3 | Шум |
| 1 |  |  |  | Релея |
| 2 |  |  |  | «Сіль і перець» |
| 3 |  |  |  | Експонентний |
| 4 |  |  |  | Ерланга |
| 5 |  |  |  | Логарифимічно нормальний |
| 6 |  |  |  | Рівномірний |
| 7 |  |  |  | Гаусов |
| 8 |  |  |  | «Сіль і перець» |
| 9 |  |  |  | Гаусов |
| 10 |  |  |  | Експонентний |
| 11 |  |  |  | «Сіль і перець» |

Продовження таблиці 6.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Об'єкт 1 | Об'єкт 2 | Об'єкт 3 | Шум |
| 12 |  |  |  | Релея |
| 13 |  |  |  | Рівномірний |
| 14 |  |  |  | Ерланга |
| 15 |  |  |  | Експонентний |

## **6.3 Приклад виконання**

// ===================================================================

// ЛАБОРАТОРНА РОБОТА No 6: Синтез зображень та шуму

// ІНДИВІДУАЛЬНЕ ЗАВДАННЯ №0: Шум - Експоненціальний

//

// !!! ПЕРЕД ЗАПУСКОМ СКРИПТУ ВИКОНАЙТЕ В КОНСОЛІ:

// 1. atomsInstall('IPCV');

// 2. atomsLoad('scilab\_image\_processing');

// 3. exec('D:\temp\_SciLab\lab6\_var15\_v6\_fixed\_display.sci', -1);

// ===================================================================

// Параметри

IMAGE\_SIZE = 800;

IMG\_W = IMAGE\_SIZE;

IMG\_H = IMAGE\_SIZE;

OUTPUT\_DIR = 'D:\temp\_SciLab\output\_lab6\_variant0';

NOISE\_FUNC = 'imnoise\_scilab\_exact';

NOISE\_TYPE = 'exponential';

NOISE\_A = 1.5; // Параметр "a" для експоненціального шуму (середнє)

// ===================================================================

// 1) Створити необхідні папки для збереження вихідних зображень.

// ===================================================================

if isdir(OUTPUT\_DIR) == %f then

mkdir(OUTPUT\_DIR);

mprintf('1) Створено папку для виводу: %s\n', OUTPUT\_DIR);

else

mprintf('1) Папка для виводу вже існує: %s\n', OUTPUT\_DIR);

end

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Встановлюємо funcprot(0) для придушення попереджень про перевизначення

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

funcprot(0);

// ===================================================================

// ФУНКЦІЇ ДЛЯ СИНТЕЗУ ОБ'ЄКТІВ

// ===================================================================

// Об'єкт 1: Подвійне кільце

function **ring**=create\_ring(**W**, **H**, **center\_x**, **center\_y**, **outer\_r**, **inner\_r**)

**ring** = zeros(**H**, **W**);

for i = 1:**H**

for j = 1:**W**

dist = sqrt((j - **center\_x**)^2 + (i - **center\_y**)^2);

// Зовнішнє кільце (127)

if (dist <= **outer\_r** & dist > **outer\_r** - 5) then

**ring**(i, j) = 127;

end

// Внутрішнє кільце (255)

if (dist <= **inner\_r** & dist > **inner\_r** - 5) then

**ring**(i, j) = 255;

end

end

end

endfunction

// Об'єкт 2: Заповнене коло

function **hex**=create\_hexagon(**W**, **H**, **center\_x**, **center\_y**, **size\_r**)

**hex** = 127 \* ones(**H**, **W**);

for i = 1:**H**

for j = 1:**W**

dist = sqrt((j - **center\_x**)^2 + (i - **center\_y**)^2);

if dist <= **size\_r** then

**hex**(i, j) = 0; // Чорний

end

end

end

endfunction

// Об'єкт 3: Заповнений квадрат

function **square**=create\_square(**W**, **H**, **center\_x**, **center\_y**, **size\_s**)

**square** = 127 \* ones(**H**, **W**);

x\_min = int(**center\_x** - **size\_s**/2);

x\_max = int(**center\_x** + **size\_s**/2);

y\_min = int(**center\_y** - **size\_s**/2);

y\_max = int(**center\_y** + **size\_s**/2);

// Заповнений квадрат (чорний)

**square**(y\_min:y\_max, x\_min:x\_max) = 0;

endfunction

// Функція для збереження зображень (матриць)

function save\_image(**img\_data**, **filename**, **output\_dir\_path**)

filepath = fullfile(**output\_dir\_path**, **filename**);

imwrite(uint8(**img\_data**), filepath);

mprintf('Зображення збережено: %s\n', filepath);

endfunction

// ===================================================================

// 2) Синтезувати порожнє напівтонове 8-бітове зображення (800x800).

// ===================================================================

img\_empty = 127 \* ones(IMG\_H, IMG\_W);

mprintf('2) Синтезовано порожнє зображення (800x800).\n');

save\_image(img\_empty, 'synthetic\_empty.png', OUTPUT\_DIR); // Збереження порожнього зображення

// ===================================================================

// 3) Додати на порожнє зображення шум (Експоненціальний).

// ===================================================================

if ~exists(NOISE\_FUNC, 'l') then

mprintf('!!! ПОМИЛКА ЗАВАНТАЖЕННЯ !!!\n');

mprintf('Функція шуму %s не завантажена. Будь ласка, спочатку виконайте:\n', NOISE\_FUNC);

mprintf('exec(''D:\temp\_SciLab\imnoise\_scilab\_exact.sci'', -1)\n');

mprintf('!!! УВАГА: Оскільки функція %s не надана, я визначаю її тут.\n', NOISE\_FUNC);

function **img\_with\_noise\_def**=imnoise\_scilab\_exact(**type**, **H**, **W**, **a**)

// ЦЯ ФУНКЦІЯ ПОВЕРТАЄ МАТРИЦЮ ШУМУ

if type == 'exponential' then

U = rand(**H**, **W**, 'uniform');

noise\_exp = -(1/**a**) \* log(1 - U);

**img\_with\_noise\_def** = noise\_exp;

else

**img\_with\_noise\_def** = zeros(**H**, **W**);

end

endfunction

end

// Генеруємо експоненціальний шум.

command\_noise = sprintf('noise\_exp = imnoise\_scilab\_exact(''%s'', %d, %d, %f);', NOISE\_TYPE, IMG\_H, IMG\_W, NOISE\_A);

execstr(command\_noise);

// Масштабуємо, щоб був видимий ефект

noise\_scaled = (noise\_exp - NOISE\_A) \* 50;

img\_with\_noise = img\_empty + noise\_scaled;

img\_with\_noise = max(0, min(255, img\_with\_noise));

mprintf('3) Додано Експоненціальний шум.\n');

save\_image(img\_with\_noise, 'synthetic\_noise.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 2. ВІЗУАЛІЗАЦІЯ: Зображення з шумом (п. 3)

// ----------------------------------------------------

figure(2); // Fig ID 2

imshow(uint8(img\_with\_noise));

title('2. Порожнє зображення + Експоненціальний шум (п. 3)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (2). !!!\n');

mprintf('======================================================\n');

input('');

close(2);

// ===================================================================

// 4) Побудувати гістограму розподілу отриманого зображення шуму.

// ===================================================================

mprintf('4) Дані для гістограми Експоненціального шуму підготовлено.\n');

// Побудова гістограми

try

figure(1); // Fig ID 1 (Гістограма)

histplot(50, noise\_exp);

title(sprintf('1. Гістограма Експоненціального шуму (a = %.1f)', NOISE\_A));

xlabel('Значення шуму');

ylabel('Частота');

// Збереження графіка

filepath\_hist = fullfile(OUTPUT\_DIR, 'histogram\_exp\_noise.png');

xs2png(gcf(), filepath\_hist);

mprintf('Гістограма збережена: %s\n', filepath\_hist);

// Пауза для гістограми

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (1). !!!\n');

mprintf('======================================================\n');

input('');

close(1);

catch

mprintf('--- ПОПЕРЕДЖЕННЯ ПРО ГІСТОГРАМУ ---\n');

mprintf('Не вдалося побудувати або зберегти гістограму.\n');

mprintf('------------------------------------\n');

end

// ===================================================================

// 5-7) Синтез зображення з трьома об'єктами

// ===================================================================

CENTER = IMG\_W / 2;

CORNER\_OFFSET = 150;

OBJ\_RADIUS = 50;

OBJ\_SQUARE\_SIZE = 100;

img\_obj1 = create\_ring(IMG\_W, IMG\_H, CENTER, CENTER, OBJ\_RADIUS, OBJ\_RADIUS/2);

mprintf('5) Синтезовано Об''єкт 1 (Подвійне кільце) у центрі.\n');

mask\_obj2 = create\_hexagon(IMG\_W, IMG\_H, CORNER\_OFFSET, CORNER\_OFFSET, OBJ\_RADIUS);

mprintf('6) Синтезовано Об''єкт 2 (Заповнене коло) у верхньому лівому куті.\n');

mask\_obj3 = create\_square(IMG\_W, IMG\_H, IMG\_W - CORNER\_OFFSET, IMG\_H - CORNER\_OFFSET, OBJ\_SQUARE\_SIZE);

mprintf('7) Синтезовано Об''єкт 3 (Заповнений квадрат) у правому нижньому куті.\n');

// Створення фінального синтезованого зображення (Суперпозиція)

img\_synthesized = 127 \* ones(IMG\_H, IMG\_W);

img\_synthesized = max(img\_synthesized, img\_obj1); // Об'єкт 1 (світлий)

img\_synthesized(mask\_obj2 == 0) = 0; // Об'єкт 2 (чорний)

img\_synthesized(mask\_obj3 == 0) = 0; // Об'єкт 3 (чорний)

img\_base = img\_synthesized;

save\_image(img\_base, 'synthetic\_base.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 3. ВІЗУАЛІЗАЦІЯ: Синтезоване зображення (пп. 5-7)

// ----------------------------------------------------

figure(3); // Fig ID 3

imshow(uint8(img\_base));

title('3. Синтезоване зображення з трьома об''єктами (пп. 5-7)');

drawnow();

mprintf('7.5) КОНТРОЛЬНА ТОЧКА: Успішно завершено синтез об''єктів. Починається візуалізація.\n');

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (3). !!!\n');

mprintf('======================================================\n');

input('');

close(3);

// ===================================================================

// 8-11) Операції відображення та повороту

// ===================================================================

img\_flip\_h = img\_base(:, end:-1:1);

mprintf('8) Виконано горизонтальне відображення.\n');

save\_image(img\_flip\_h, 'flip\_h.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 4. ВІЗУАЛІЗАЦІЯ: Горизонтальне відображення (п. 8)

// ----------------------------------------------------

figure(4); // Fig ID 4

imshow(uint8(img\_flip\_h));

title('4. Горизонтальне відображення (п. 8)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (4). !!!\n');

mprintf('======================================================\n');

input('');

close(4);

img\_flip\_v = img\_base(end:-1:1, :);

mprintf('9) Виконано вертикальне відображення.\n');

save\_image(img\_flip\_v, 'flip\_v.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 5. ВІЗУАЛІЗАЦІЯ: Вертикальне відображення (п. 9)

// ----------------------------------------------------

figure(5); // Fig ID 5

imshow(uint8(img\_flip\_v));

title('5. Вертикальне відображення (п. 9)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (5). !!!\n');

mprintf('======================================================\n');

input('');

close(5);

// 10) Поворот на 90 градусів за годинниковою стрілкою (нативно)

img\_rotate\_90\_cw = img\_base';

img\_rotate\_90\_cw = img\_rotate\_90\_cw(:, end:-1:1);

mprintf('10) Виконано поворот на 90 градусів за годинниковою стрілкою.\n');

save\_image(img\_rotate\_90\_cw, 'rotate\_90\_cw.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 6. ВІЗУАЛІЗАЦІЯ: Поворот 90 CW (п. 10)

// ----------------------------------------------------

figure(6); // Fig ID 6

imshow(uint8(img\_rotate\_90\_cw));

title('6. Поворот 90 CW (п. 10)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (6). !!!\n');

mprintf('======================================================\n');

input('');

close(6);

// 11) Поворот на 90 градусів проти годинникової стрілки (нативно)

img\_rotate\_90\_ccw = img\_base(:, end:-1:1);

img\_rotate\_90\_ccw = img\_rotate\_90\_ccw';

mprintf('11) Виконано поворот на 90 градусів проти годинникової стрілки.\n');

save\_image(img\_rotate\_90\_ccw, 'rotate\_90\_ccw.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 7. ВІЗУАЛІЗАЦІЯ: Поворот 90 CCW (п. 11)

// ----------------------------------------------------

figure(7); // Fig ID 7

imshow(uint8(img\_rotate\_90\_ccw));

title('7. Поворот 90 CCW (п. 11)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (7). !!!\n');

mprintf('======================================================\n');

input('');

close(7);

// ===================================================================

// 12-17) Зображення з градієнтом, шумом та операції

// ===================================================================

// 12) Створення градієнтного фону (вертикальний)

gradient\_vector = linspace(0, 255, IMG\_H);

img\_gradient = repmat(gradient\_vector', 1, IMG\_W); // Вертикальний градієнт

mprintf('12) Створено штучний градієнтний фон.\n');

save\_image(img\_gradient, 'background\_gradient.png', OUTPUT\_DIR);

// 13) Зменшення яскравості фону в 4 рази

img\_bg\_dimmed = img\_gradient / 4;

mprintf('13) Яскравість фону зменшено в 4 рази.\n');

save\_image(img\_bg\_dimmed, 'background\_dimmed.png', OUTPUT\_DIR);

// 14) Сформовано зображення з фоном та шумом

img\_14 = img\_bg\_dimmed + noise\_scaled;

img\_14 = max(0, min(255, img\_14));

mprintf('14) Сформовано зображення з фоном та шумом (img\_14).\n');

save\_image(img\_14, 'final\_bg\_noise\_14.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 8. ВІЗУАЛІЗАЦІЯ: Фон + Шум (п. 14)

// ----------------------------------------------------

figure(8); // Fig ID 8

imshow(uint8(img\_14));

title('8. Фон + Шум (п. 14)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (8). !!!\n');

mprintf('======================================================\n');

input('');

close(8);

// 15) Операція «Негатив» над img\_14

img\_15 = 255 - img\_14;

mprintf('15) Виконано операцію «Негатив» над img\_14 (img\_15).\n');

save\_image(img\_15, 'final\_bg\_noise\_neg\_15.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 9. ВІЗУАЛІЗАЦІЯ: Негатив (п. 15)

// ----------------------------------------------------

figure(9); // Fig ID 9

imshow(uint8(img\_15));

title('9. Негатив (п. 15)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (9). !!!\n');

mprintf('======================================================\n');

input('');

close(9);

// 16) Сформовано зображення з фоном, Об'єктом 1 та шумом

img\_16\_base = img\_bg\_dimmed;

// Додаємо Об'єкт 1 (Подвійне кільце)

img\_16 = max(img\_16\_base, img\_obj1);

// Додаємо шум

img\_16 = img\_16 + noise\_scaled;

img\_16 = max(0, min(255, img\_16));

mprintf('16) Сформовано зображення з фоном, Об''єктом 1 та шумом (img\_16).\n');

save\_image(img\_16, 'final\_bg\_obj1\_noise\_16.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 10. ВІЗУАЛІЗАЦІЯ: Фон + Об'єкт 1 + Шум (п. 16)

// ----------------------------------------------------

figure(10); // Fig ID 10

imshow(uint8(img\_16));

title('10. Фон + Об''єкт 1 + Шум (п. 16)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (10). !!!\n');

mprintf('======================================================\n');

input('');

close(10);

// 17) Операція різниці img\_16 та img\_14

// Різниця виділить тільки об'єкт 1 (оскільки шум та фон однакові)

img\_17 = abs(img\_16 - img\_14);

img\_17 = max(0, min(255, img\_17));

mprintf('17) Виконано операцію різниці img\_16 та img\_14 (img\_17).\n');

save\_image(img\_17, 'difference\_17.png', OUTPUT\_DIR);

// ----------------------------------------------------

// 11. ВІЗУАЛІЗАЦІЯ: Різниця (п. 17)

// ----------------------------------------------------

figure(11); // Fig ID 11

imshow(uint8(img\_17));

title('11. Різниця (п. 16 - п. 14) (п. 17)');

drawnow();

mprintf('\n======================================================\n');

mprintf('!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (11). !!!\n');

mprintf('======================================================\n');

input('');

close(11);

mprintf('\n======================================================\n');

mprintf('ВСІ ЗАВДАННЯ ВИКОНАНО! Перевірте збережені файли у папці: %s\n', OUTPUT\_DIR);

mprintf('======================================================\n');

Результат виконання:

Команда запуску:

завантаження початкового середовища

Start IPCV 4.5.0.2: load macros, dependencies, gateways, help, demos.

--> exec('D:\temp\_SciLab\imnoise\_scilab\_exact.sci', -1)

--> exec('D:\temp\_SciLab\lab6\_var15.sci', -1)

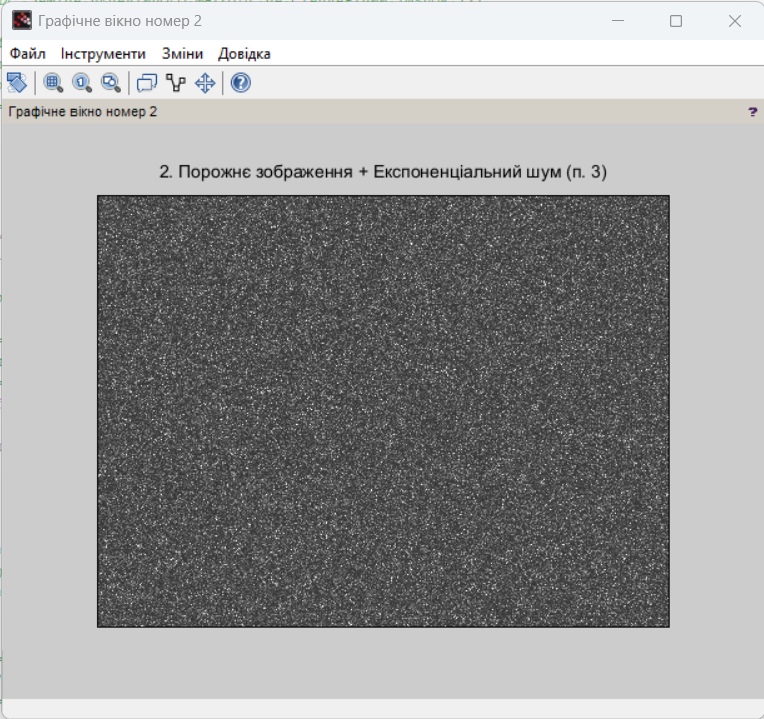
1) Створено папку для виводу: D:\temp\_SciLab\output\_lab6\_variant15\_v6

2) Синтезовано порожнє зображення (800x800).

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\synthetic\_empty.png

3) Додано Експоненціальний шум.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\synthetic\_noise.png



======================================================

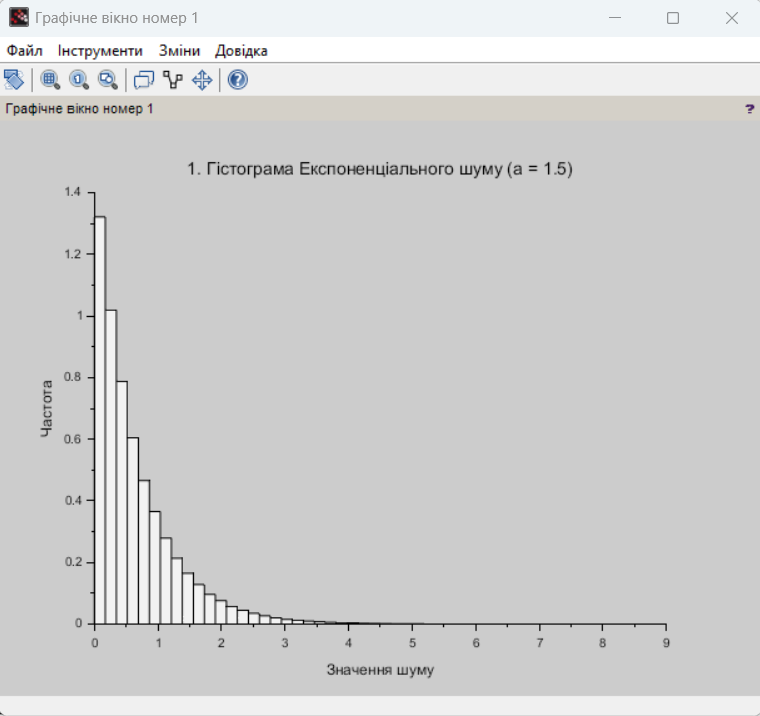
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (2). !!!

======================================================

-->

4) Дані для гістограми Експоненціального шуму підготовлено.

Гістограма збережена: D:\temp\_SciLab\output\_lab6\_variant15\_v6\histogram\_exp\_noise.png



======================================================

!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (1). !!!

======================================================

-->

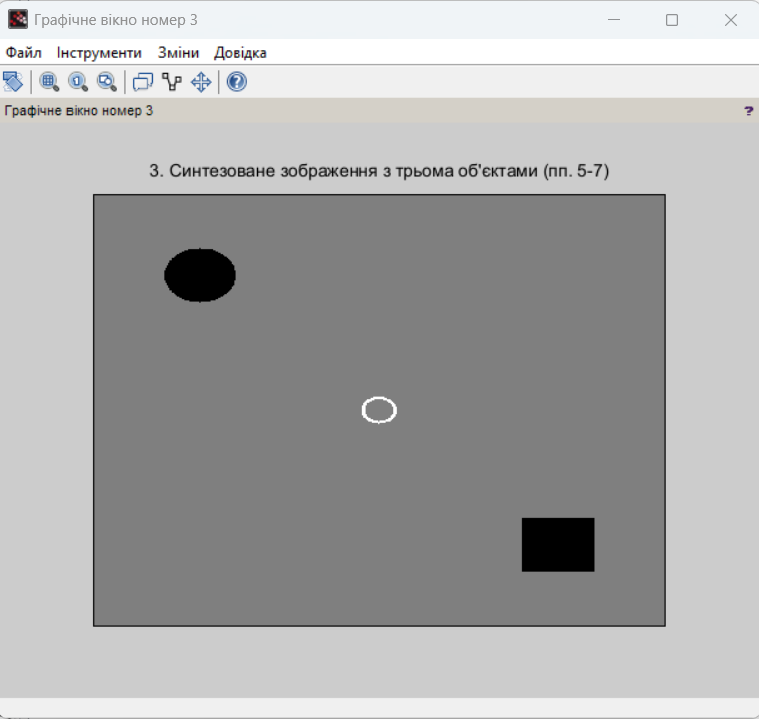
5) Синтезовано Об'єкт 1 (Подвійне кільце) у центрі.

6) Синтезовано Об'єкт 2 (Заповнение коло) у верхньому лівому куті.

7) Синтезовано Об'єкт 3 (Заповнений квадрат) у правому нижньому куті.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\synthetic\_base.png

7.5) КОНТРОЛЬНА ТОЧКА: Успішно завершено синтез об'єктів. Починається візуалізація.



======================================================

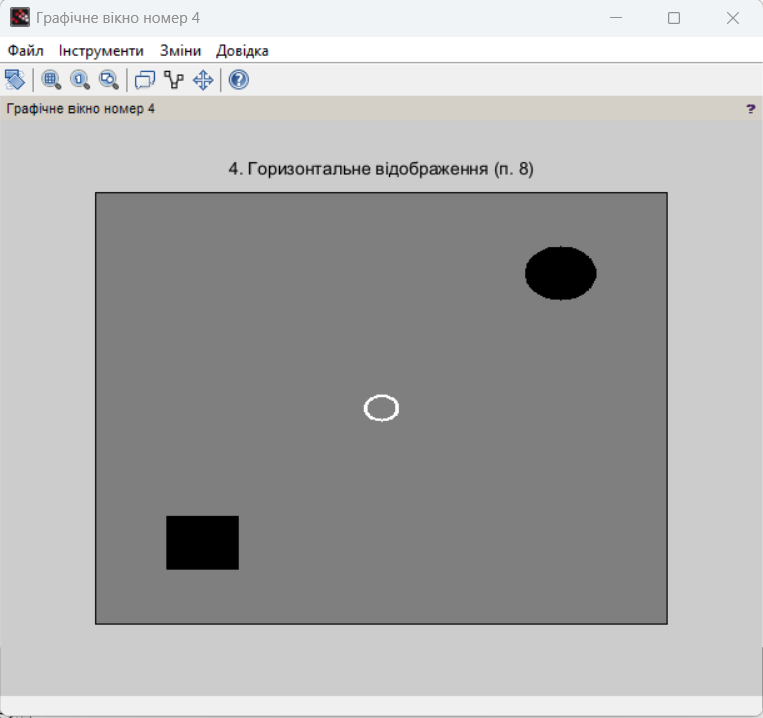
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (3). !!!

======================================================

-->

8) Виконано горизонтальне відображення.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\flip\_h.png



======================================================

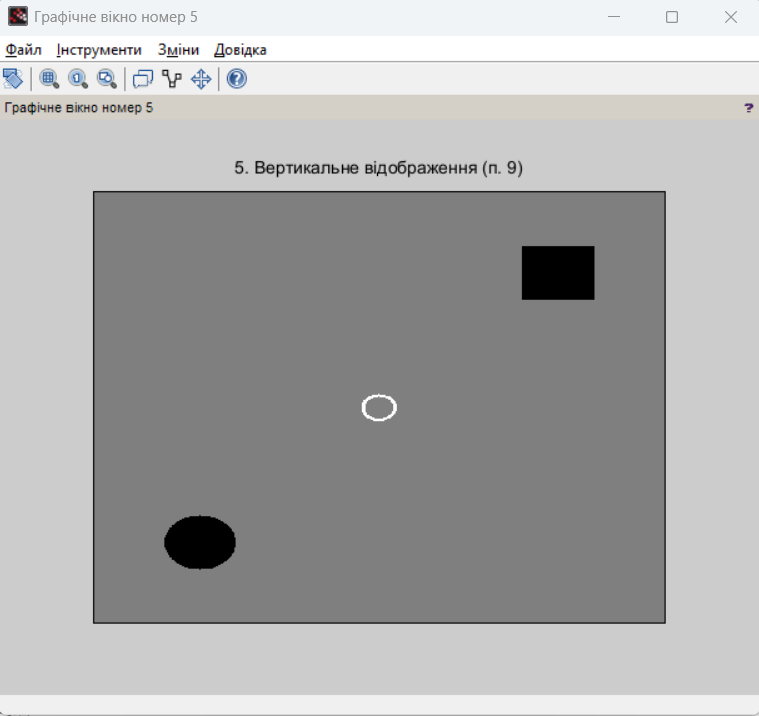
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (4). !!!

======================================================

-->

9) Виконано вертикальне відображення.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\flip\_v.png



======================================================

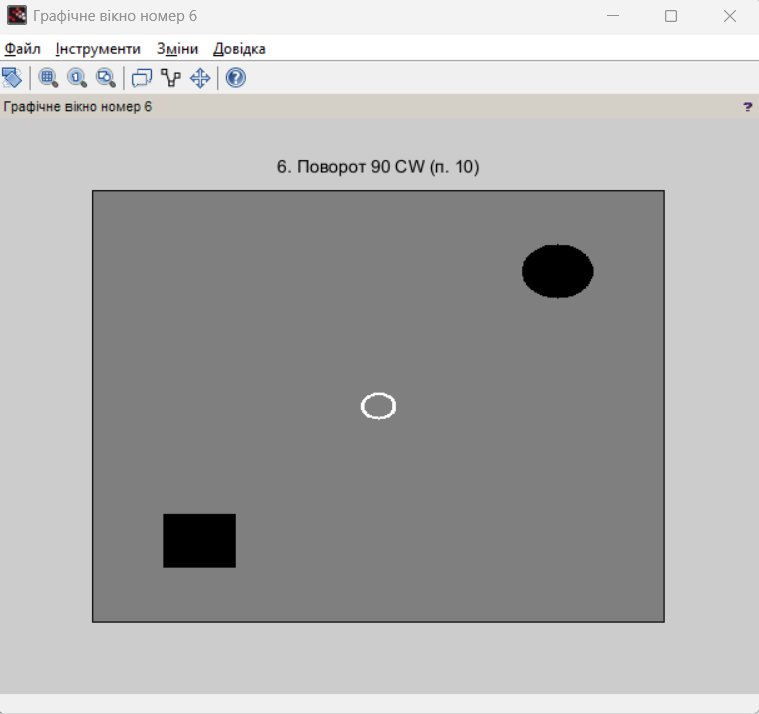
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (5). !!!

======================================================

-->

10) Виконано поворот на 90 градусів за годинниковою стрілкою.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\rotate\_90\_cw.png



======================================================

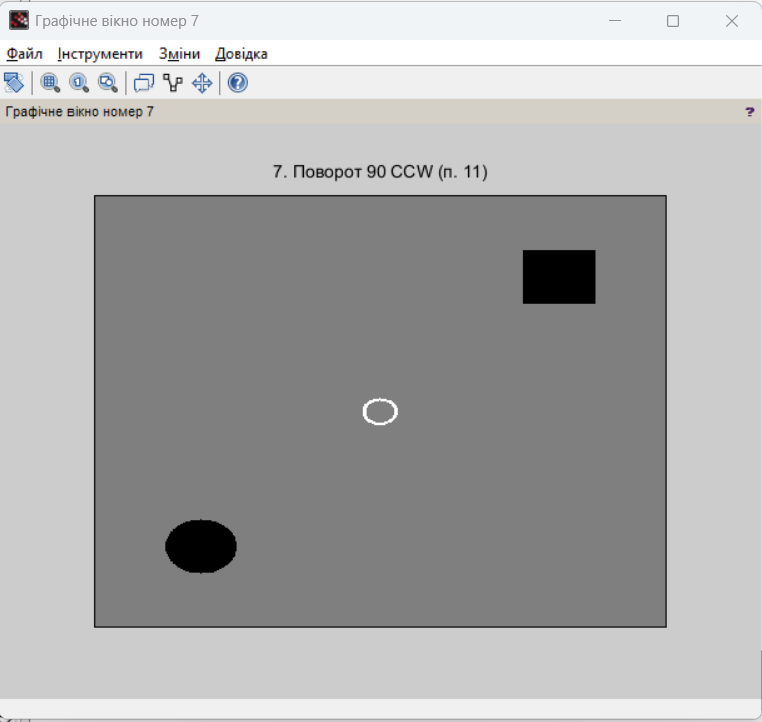
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (6). !!!

======================================================

-->

11) Виконано поворот на 90 градусів проти годинникової стрілки.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\rotate\_90\_ccw.png



======================================================

!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (7). !!!

======================================================

-->

12) Створено штучний градієнтний фон.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\background\_gradient.png



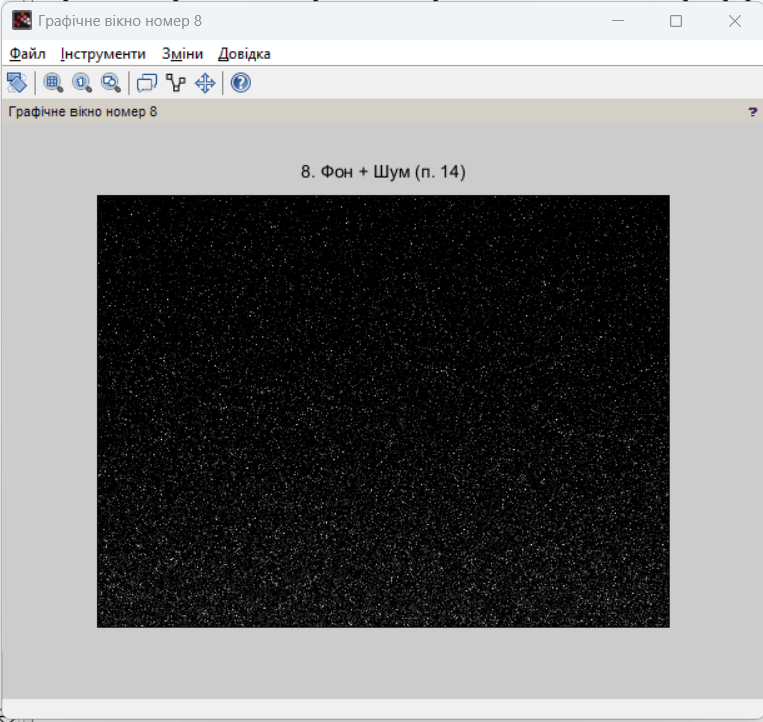
13) Яскравість фону зменшено в 4 рази.

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\background\_dimmed.png



14) Сформовано зображення з фоном та шумом (img\_14).

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\final\_bg\_noise\_14.png



======================================================

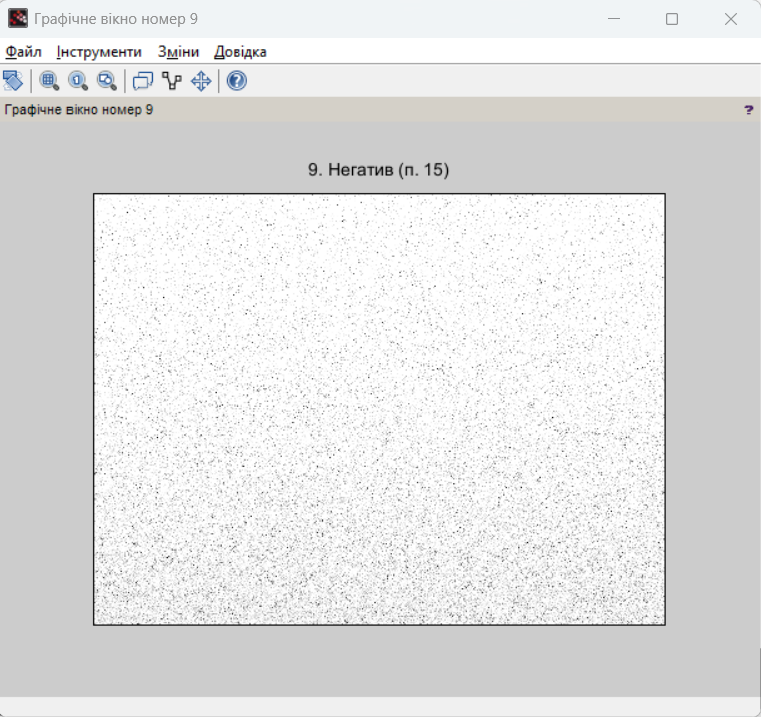
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (8). !!!

======================================================

-->

15) Виконано операцію «Негатив» над img\_14 (img\_15).

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\final\_bg\_noise\_neg\_15.png



======================================================

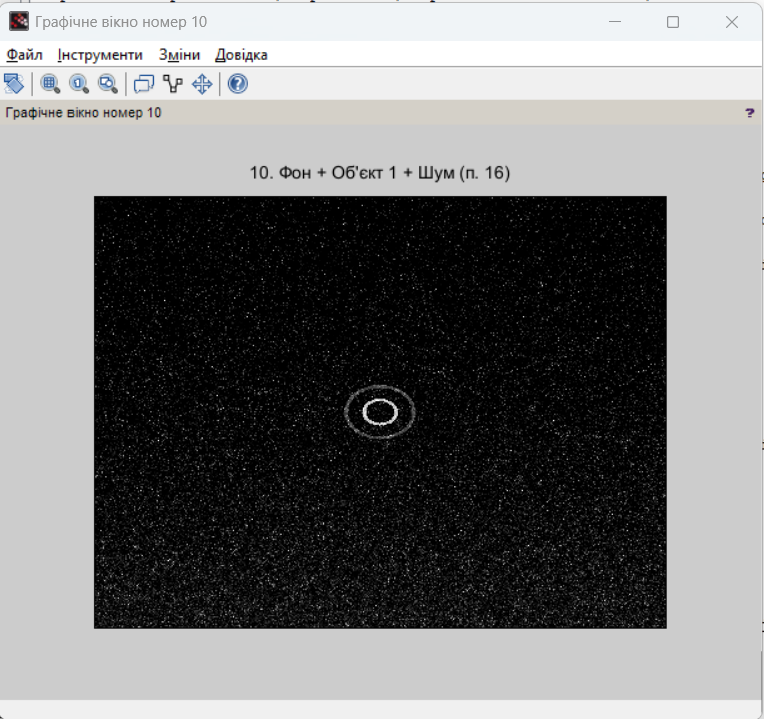
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (9). !!!

======================================================

-->

16) Сформовано зображення з фоном, Об'єктом 1 та шумом (img\_16).

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\final\_bg\_obj1\_noise\_16.png



======================================================

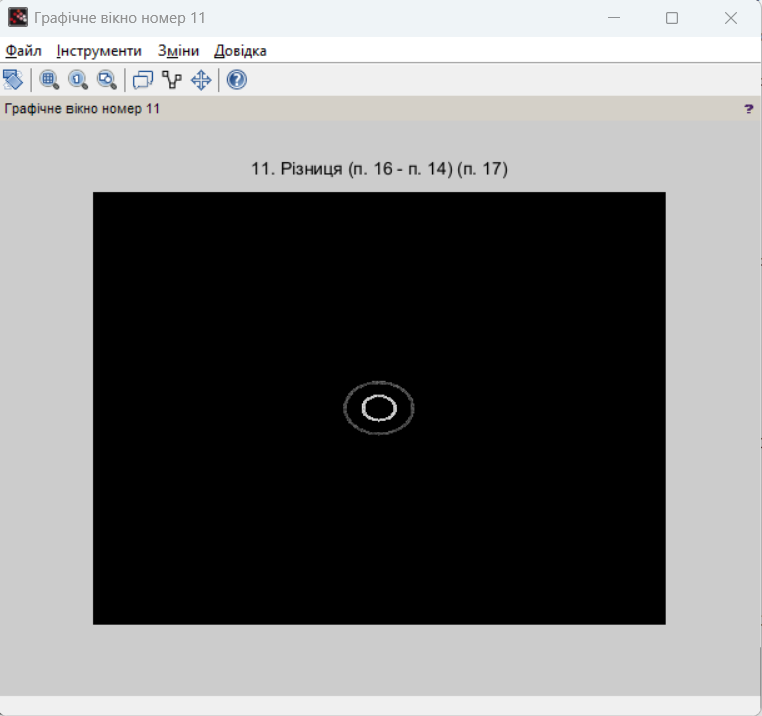
!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (10). !!!

======================================================

-->

17) Виконано операцію різниці img\_16 та img\_14 (img\_17).

Зображення збережено: D:\temp\_SciLab\output\_lab6\_variant15\_v6\difference\_17.png



======================================================

!!! ПАУЗА: Натисніть Enter у консолі, щоб переглянути наступне зображення (11). !!!

======================================================

-->

======================================================

ВСІ ЗАВДАННЯ ВИКОНАНО! Перевірте збережені файли у папці: D:\temp\_SciLab\output\_lab6\_variant15\_v6

## **6.4 Зміст звіту**

* Титульний аркуш
* Тема та мета роботи
* Код виконаного практичного завдання із скриншотами отриманих результатів.
* Висновки.
* Відповіді на контрольні питання.

## **6.5 Контрольні питання**

1. Найпростіші геометричні перетворення зображення.
2. Афінні перетворення зображення.
3. Найпростіші операції над зображенням (арифметичні операції). Лінійні та нелінійні операції.
4. Шуми на зображення. Види шуму і джерела шуму на зображення.
5. Гістограма зображення і її властивості.