**Київский національний університет імені Тараса Шевченка**

**Факультет компьютерних наук та кібернетики**

**Моделювання Систем**

**Лабораторна робота 3**

**Звіт**

**Варіант 5**

**Виконав:**

студент групи ІС-31

Дьяченко Максим

**Київ-2018**

**Умова:**

Для послідовності вхідних сигналів ,  та вихідних сигналів ,  знайти оператор  перетворення вхідного сигналу у вихідний.

Будемо шукати математичну модель оператора об'єкту в класі лінійних операторів

, . (1)

Невідома матриця  математичної моделі об'єкту розмірності . Систему (1) запишемо у матричній формі

,

або

, (2)

де  – матриця вхідних сигналів розмірності ,  – матриця вихідних сигналів розмірності .

Матрицю  будемо інтерпретувати як двовимірне вхідне зображення, а матрицю  вихідне зображення.

Тоді

,

де матриця

,

розмірності , .

Варіанти вхідних на вихідних сигналів, для яких потрібно побудувати лінійний оператор перетворення вхідного сигналу у вихідний:

Код програми:

**Grevil.m:**

function [ RES ] = Grevil( X )

m = size(X, 1);

a = X(1, :).';

A(1, :) = a;

if((a.' \* a) == 0)

RES = a;

else

RES = a / (a.' \* a);

end

e = 1^-6;

for k=2:m

a = X(k, :).';

AA = RES \* A;

one = ones(1, size(AA, 1));

Z = diag(one) - AA;

atZa = a.' \* Z \* a;

A(k, :) = a;

if atZa > 0

RES = RES - (Z \* a \* a.' \* RES) / atZa;

RES(:, k) = (Z \* a) / atZa;

end

if abs(atZa) <= e

R = RES \* RES.';

atRa = a.' \* R \* a;

RES = RES - (R \* a \* a.' \*RES) / (1 + atRa);

RES(:, k) = (R \* a) / (1 + atRa);

end

end

end

**main.m:**

clc

clear

X = imread('X3.bmp');

Y = imread('Y3.bmp');

doubleX = double(X);

doubleY = double(Y);

doubleX = [doubleX; ones(1, size(doubleX, 2))];

[m,n] = size(doubleX);

%Grevill

grevX = Grevil(doubleX);

XX\_grev = doubleX \* grevX;

one = ones(1, size(XX\_grev, 1));

Z\_grev = diag(one) - XX\_grev;

V = rand(size(doubleY, 1), size(grevX, 2));

%Mura-Penrouse

d = 10;

X1=[];

X2=[];

eps = 0.002;

difX1X2 = 200;

while ( difX1X2 > eps )

if( m > n)

X1 = inv(doubleX' \* doubleX - d \* d \* eye(n)) \* doubleX';

else

X1 = doubleX' \* inv(doubleX \* doubleX' - d \* d \* eye(m));

end

d = d / 2;

if( m > n)

X2 = inv(doubleX' \* doubleX - d \* d \* eye(n)) \* doubleX';

else

X2 = doubleX' \* inv(doubleX \* doubleX' - d \* d \* eye(m));

end

difX1X2 = norm(X1 - X2);

end

mpX = X1;

Z\_mp = eye(m) - doubleX \* mpX;

A\_MP = doubleY \* mpX + V \* Z\_mp';

RES\_MP = A\_MP \* doubleX;

RES\_MP\_uint = uint8(RES\_MP);

A\_Grev = doubleY \* grevX+ V \* Z\_grev.';

RES\_Grev = A\_Grev \* doubleX;

RES\_Grev\_uint = uint8(RES\_Grev);

%resulting pictures

figure();

imshow(Y);

title('Input data : Y');

figure();

imshow(RES\_Grev\_uint);

title('Grevill method');

figure();

imshow(RES\_MP\_uint);

title('Moore-Penrouse method');

Результатом роботи програми є :

  