Task 4 - Filtration in frequency domain.

The aim of the task is to learn how to construct fast algorithms for calculation of discrete orthogonal transforms and how to use them in practical tasks. As an example Fourier transform is considered.

This excercise is composed of the following elements:

- Implement slow and fast discrete normal and inverse Fourier transform. During the implementation of fast variant the number of arithmetic operations as well as the amount of the needed memory should be minimized. The available variants are:
  - (T1) Normal and inverse fast Fourier transform with decimation in spatial domain.
  - (T2) Normal and inverse fast Fourier transform with decimation in frequency domain.
- Visualization of Fourier spectrum.
- Implementation of the following filters in frequency domain:
  - (F1) Low-pass filter (high-cut filter).
  - (F2) High-pass filter (low-cut filter).
  - (F3) Band-pass filter.
  - (F4) Band-cut filter.
  - (F5) High-pass filter with detection of edge direction
  - (F6) Phase modifying filter

In all the variants the special attention should be paid to implementation efficiency (reduction of needed memory, reduction of execution time).

Implementation of filters (F1) - (F5) should allow user to specify the size of the band and in case of variant (F5) also the direction of the edges. Sample masks for variant (F5) can be found **here** and **here**, and sample test images for this variant **here**, **here** and **here** :)

Mask of filter (F6) is a matrix with complex numbers *P(n, m)*, *n*=0, ..., *N*-1, *m*=0, ..., *M*-1 (for image of size *N*x*M*), where each element has a magnitude equal to 1 and the argument (phase) depends linearly on its position in the matrix. During the filtration the corresponding elements of the Fourier spectrum and of the mask shoul be multiplied. The mask *P* can be defined in the following way:

$$P(n,m) = \exp\left(j \cdot \left(\frac{-nk \cdot 2\pi}{N} + \frac{-ml \cdot 2\pi}{M} + (k+l) \cdot \pi \right) \right)$$

where *l, k* are integer numbers and should be given by the user.

The report template:

- **report_4.doc** - Microsoft Word

Instructions and remarks:

Two-dimensional transforms:

An easy method for constriction of two-dimensional discrete orthogonal transforms is thier realization using the set of one-dimensional transforms. Let*x(n, m)*, *n*=0, ..., *N*-1, *m*=0, ..., *M*-1, be the matrix of complex (real) numbers, anf led *A(p, q)* be its two-dimensional *N*x*M*-point discrete orthogonal transform. To calculate *A(p, q)* one can use:

- *M* one-dimensional *N*-point transforms of rows of matrix macierzy *x(n, m)*.
- *N* one-dimensional *M*-point transforms of columns of the matrix being the result of previous step.

Transform equations nad graphs:

- Discrete Fourier trasnform (DFT)

  One-dimensional DFT.

  Let *x(n)* for *n*=0,...., *N*-1, be a vactor of complex numbers. Then **normal** *N*-point discrete Fourier transform of vector *x(n)* is a vector *X(k)* where:

$$X(k) = DFT_N\{x(n)\} = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \text{ dla } k = 0,\ldots,N-1$$

$$\text{gdzie } W_N^{kn} = e^{-i2\pi nk/N} = \cos(2\pi nk/N) - i\sin(2\pi nk/N), i = \sqrt{-1}.$$

Fast Fourier transform (FFT) using butterfly notation:

- decimation in spatial domain

$$X(k) = X_1(k) + W_N^k X_2(k), X(k+N/2) = X_1(k) - W_N^k X_2(k), k = 0,\ldots,N/2-1,$$
$$\text{gdzie } X(k) = DFT_N\{x(n)\}, X_1(k) = DFT_{N/2}\{x(2n)\}, X_2(k) = DFT_{N/2}\{x(2n+1)\}.$$

- decimation in frequency domain

$$X(2k) = DFT_{N/2}\{x(n) + x(n+N/2)\},$$
$$X(2k+1) = DFT_{N/2}\{(x(n) - x(n+N/2))W_N^n\}, k = 0,\ldots,N/2-1.$$

Inverse *N*-point discrete Fourier transform (IDFT) *x(n)*, can be found using:

$$x(n) = IDFT_N\{X(k)\} = \frac{1}{N}\sum_{k=0}^{N-1} X(k) W_N^{-nk}, n = 0,\ldots,N-1,$$

$$\text{gdzie } W_N^{-nk} = e^{i2\pi nk/N} = \cos(2\pi nk/N) + i\sin(2\pi nk/N), i = \sqrt{-1}.$$

Inverse fast Fourier transform (IFFT) is similar to normal transform. The result must be multiplied by constant *1/N* and there is the change of the sign in W_N^{nk}. Of course:

$$x(n) \equiv IDFT_N\{DFT_N\{x(n)\}\}.$$

Two-dimensional DFT.

Let *x(n, m)*, *n*=0,..., *N*-1, *m*=0,..., *M*-1, be a matrix of complex numbers. Then **normal** two-dimensional *NxM*-point discrete Fourier transform is a complex matrix *X(p, q)*, where:

$$X(p,q) = DFT_{N*M}\{x(n,m)\} = \frac{1}{\sqrt{NM}}\sum_{n=0}^{N-1}\sum_{m=0}^{M-1} x(n,m) W_N^{np} W_M^{mq},$$

$$\text{gdzie } W_N^{nk} = e^{-i2\pi nk/N}, p = 0,\ldots,N-1, q = 0,\ldots,M-1, i = \sqrt{-1}.$$

Inverse two-dimensional *NxM*-point DFT (IDFT) is a complex matrix *x(n, m)* where:

$$x(n,m) = IDFT_{N*M}\{X(p,q)\} = \frac{1}{\sqrt{NM}}\sum_{p=0}^{N-1}\sum_{q=0}^{M-1} X(p,q) W_N^{-np} W_M^{-mq},$$

$$\text{gdzie } W_N^{-nk} = e^{i2\pi nk/N}, n = 0,\ldots,N-1, m = 0,\ldots,M-1, i = \sqrt{-1}.$$

Of course, as it was in one-dimensional case:

$$x(n,m) \equiv IDFT_{N*M}\{DFT_{N*M}\{x(n,m)\}\}.$$

- Graphs:

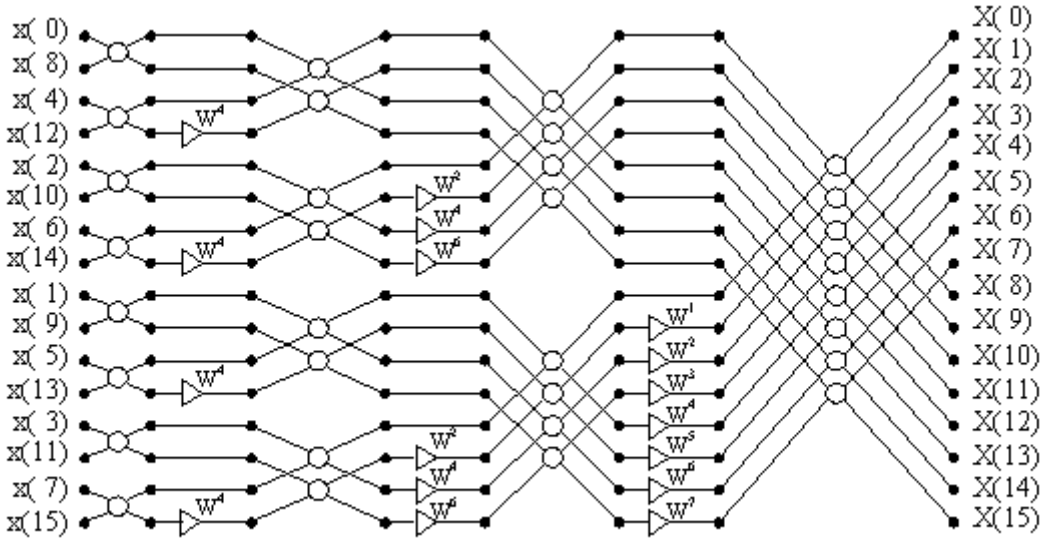

Fig. 1. Graph for 16-point FFT algorithm (decimation in spatial domain)

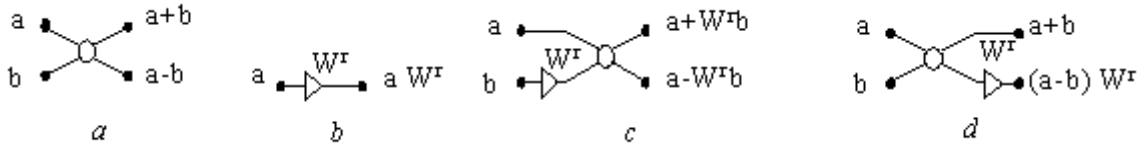

Fig. 2. Notation for butterfly graphs of FFT algorithm

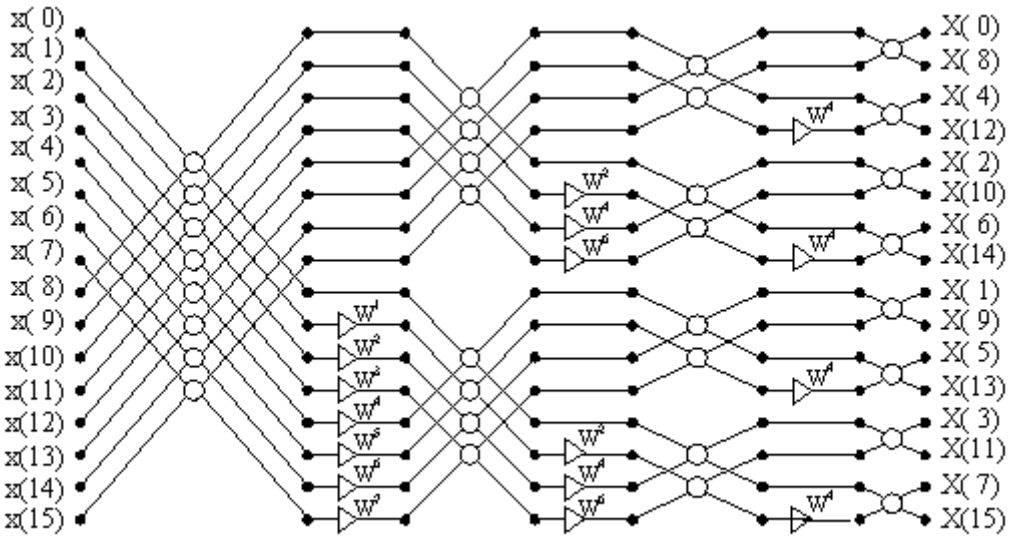

Fig. 3. Graph for 16-point FFT algorithm (decimation in frequency domain)