Lab 6 IPT 2022-2023

Applications of the wavelet transform for image filtering

1. Introduction

Wavelet transformed image filtering techniques separate for each high-frequency sub-band - $I(x, y)$ – the components due to useful signal $I_s(x, y)$ from those due to noise $I_{zg}(x, y)$

$$I(x, y) = I(\lambda) = \{I_s(x, y)\} \cup \{I(_{zg}x, y)\} \quad (1)$$

According to the wavelet formalism all detail coefficients associated with a DWT decomposition with all detail coeficients in absolute value to a threshold λ are considered as a useful signal - $I_s(x, y)$ - and all other coefficients of detail are considered to come from noise $I_{zg}(x, y)$.
DWT-based *hard* image filtering techniques involve the cancellation of all noise coefficients and the retention of all the other coefficients from the useful signal.

Wavelet-based image filtering algorithms use some predefined steps:
- noise level estimation;
- estimation of the separation threshold between the two classes of coefficients λ.

The most used techniques, *VisuShrink* and *BayesShrink* start from the frequency band *HH...H* and estimate the noise level using the *Mean Absolute Deviation (MAD)* defined by:

$$MAD = \frac{mediana(|I_{HH}|)}{0.6745} \quad (2)$$

The separation threshold in the VisuShrink model is estimated on the basis of the following equation:

$$\lambda = MAD \cdot \sqrt{2\lg N_{SB}} \quad (3)$$

with N_{SB} the total number of detail coefficients in the considered subband.

Even if the threshold is estimated only from this frequency band, VisuShrink wavelet filtration involves its use in comparisons in all frequency subbands – except *LLL...L*. This threshold is recalculated for each level of decomposition to which the filtration operation applies.

The purpose of the lab consists in the study and implementation of a of a VisuShrink filtering method and its comparison with other types of classical image filtering methods.

2. VisuShrink algorithm implementation in VisualDSP ++

The skeleton of the application that will be used to implement the VisuShrink algorithm corresponds to an implementation of the direct and inverse DWT transforms through Haar filter banks. The Visual DSP++ project available on the discipline website includes all the necessary functions:

float Estimare_prag(float * intrare, float * hh, int latime, int inaltime) - function that allows the comparison threshold computation according to the formulas (2) și (3)

void sortare(float * valori, long numar_el) – function that allows you sorting list of values of a certain length in descending order – both transmitted as parameters in the function call

void comparare_prag(double * intrare, int lim_jos_x ,int lim_sus_x,int lim_jos_y,int lim_sus_y , double prag) – function that allows comparing the values of a region in an image with a threshold. The region to be analysed is specified by the limits on x and y called suggestively *lim_jos(sus)_x(y)*.

3. Exercises

1) Analyze how the above functions are implemented (appendix 1).
2) Modify the project *Filtrare_wavelet* for allowing VisuShrink-based filtering of an image
3) Using the results from 2, process the image **lena_zg.bmp;** analyse the obtained results and indicate what is the effect of wavelet filtering on detail coefficients . Save the result obtained in the current directory under the name **lena_wavelet.bmp.** Which are the values corresponding to 2 and 3?
4) Repeat 3) and filter the **lena_wavelet.bmp.** Which is the effect observable on the output image?
5) Change the comparison condition ***if (fabs(intrare [i+j*dimx])<prag)***
   with ***if (fabs(intrare [i+j*dimx])>0)***
   Run the app and explain the effect of the new condition by analysing the effects on the image obtained.
6) Propose implementations in another VisualDSP project for a mean and median filter (N=25). Process the same image from point 3 and compare the obtained results
7) Outline the changes to be made to the project *Filtrare_wavelet* in order to allow the algorithm to be applied on multiple levels.

Bibliografie

[1] R. Terebes – Tehnologii de prelucrare a informației – notițe de curs 2022-2023, http://ares.utcluj.ro/tpi

Appendix 1. Functions used in the implementation of the VisuShrink algorithm

```
float Estimare_prag(float* intrare, float * hh_abs,  int  latime, int  inaltime)
{
	float val_ret;
	long i,j;

	for (i=latime/2;i<latime;i++)
		for (j=inaltime/2;j<inaltime;j++)
		{
			if (intrare[i+j*latime]<0)
				hh_abs[i-latime/2+(j-inaltime/2)*latime/2]=-intrare[i+j*latime];
			else
				hh_abs[i-latime/2+(j-inaltime/2)*latime/2]=intrare[i+j*latime];
		}

	sortare(hh_abs,  latime*inaltime/4-1);

	val_ret=hh_abs[latime*inaltime/8-1]/0.6745;
	val_ret=val_ret*sqrt(2*log10(inaltime*latime/4));

return val_ret;
}

void sortare(float * valori, int numar_el)
{
	int i,j;
	double curent;
	for (i=0;i<numar_el;i++)
	{
		curent=valori[i];
		j=i-1;
		while (j>=0 && curent>valori[j])
		{
			valori[j+1]= valori[j];
			j=j-1;
		}
	valori[j+1]=curent;

	}
}

void comparare_prag(float * intrare, int lim_jos_x ,int lim_sus_x,int lim_jos_y,int lim_sus_y ,
int dimx, int dimy, float prag)
{
	int i,j;
	for (i=lim_jos_x;i<lim_sus_x;i++)
		for (j=lim_jos_y;j<lim_sus_y;j++)
			if (fabs(intrare [i+j*dimx])<prag)
				intrare[i+j*dimx]=0.0;

}
```