PROCEEDINGS

of the Union of Scientists - Ruse

Book 5

Mathematics, Informatics and Physics

Volume 9, 2012

RUSE

The Ruse Branch of the Union of Scientists in Bulgaria was founded in 1956. Its first Chairman was Prof. Stoyan Petrov. He was followed by Prof. Trifon Georgiev, Prof. Kolyo Vasilev, Prof. Georgi Popov, Prof. Mityo Kanev, Assoc. Prof. Boris Borisov, Prof. Emil Marinov, Prof. Hristo Beloev. The individual members number nearly 300 recognized scientists from Ruse, organized in 13 scientific sections. There are several collective members too – organizations and companies from Ruse, known for their success in the field of science and higher education, or their applied research activities. The activities of the Union of Scientists – Ruse are numerous: scientific, educational and other humanitarian events directly related to hot issues in the development of Ruse region, including its infrastructure, environment, history and future development; commitment to the development of the scientific organizations in Ruse, the professional development and growth of the scientists and the protection of their individual rights.

The Union of Scientists – Ruse (US – Ruse) organizes publishing of scientific and popular informative literature, and since 1998 – the "Proceedings of the Union of Scientists- Ruse".

BOOK 5

"MATHEMATICS, INFORMATICS AND PHYSICS"

VOLUME 9

CONTENTS

SOFTWARE MODULE FOR SPECTRAL ANALYSIS OF AUDIO SIGNALS

Rumen Rusev, Ana Kaneva

Angel Kanchev University of Ruse

Abstract: *This paper describes the development of a library that realizes spectral analysis of sound signals, using standard audio interface of a personal computer. In the realization are discussed the basic settings of the signals theory: digitization of analog signal, Fourier series decomposition, phase-frequency characteristics of signal and the way its spectrum is visualized.*

Keywords: *spectral analysis, digitization of sound signals, digital processing of signals, spectrum visualization*

INTRODUCTION

Signals are functions of one or more independent variables and usually contain information about the behaviour or the nature of the phenomena around us. The analysis of those signals can help for the better understanding and the process of studying the essence of those phenomena.

The exceptionally rapid progress of the contemporary computers make signals more attractive, because they have continuous (analogue) character when digitized and can easily be stored and processed with the help of universal computers or other specialized digital devices.

OVERVIEW OF THE BASIC PRINCIPLES OF THE DIGITAL SIGNAL PROCESSING

The signal is a physical process that exists in time and can be represented as its function s(t). It is known [2, 3, 5], that a random signal can be represented as an infinite sum of sine signals – Fourier series. The amplitude of the sine functions of the Fourier series define the frequency properties of the signal i.e. its frequency spectrum.

In order for a continuous signal to be processed in a digital system it must be sampled and quantized. A discrete signal is such a signal that exists only as discrete values distributed on the axis t moments tn (n is a whole number in the interval $(-\infty, \infty)$). The discrete signal is denoted with s(tn). It is important to underline the fact that in the average in ratio tn moments t, i.e. for tn-1 <t< tn and tn <t< tn+1, the discrete signal does not equal zero, it just doesn't exist. Usually all the moments tn are separated from the neighbour moments tn-1 and tn+1 with equal intervals Δt, i.e. tn are ordered equidistantly on the axis t. It is obvious that in those cases we can write tn = nΔt, respectively s(tn)=s(nΔt). Sometimes the continuous time interval Δt is omitted and instead of s(nΔt) we use [n], and instead of the term discrete signal – signal series. The square brackets in s[n] mean the exchange of the discrete time moments with whole positive and negative numbers including null.

Quite often the discrete signal s(nΔt) can be obtained from a continuous signal s(t) with its substitution of the last with the ordered series of its discrete values taken in moments tn = nΔt. This operation is called sampling of the continuous signal and the interval Δt – a sampling step. The so calculated signal can be used to adequately represent the initial continuous signal then and only then when the step Δt of digitization is sufficiently small. A discrete Fourier transformation (DFT) can be applied on a discrete signal. The mathematical expression is shown in the equation (1), where the result is a discrete complex function.

$$G[k]=\sum_{n=0}^{N-1} S[n]e^{-j\frac{2\pi}{N}nk}, \ \kappa \in \left[-\frac{N}{2},\frac{N}{2}-1\right] \qquad (1)$$

From (1) the frequency characteristics of the sound signal can be defined. Because of the fact that the computational complexity of the DFT is ~o(N2) in a practical implementation, a similar approach is used called Fast Fourier Transformation with computational complexity of ~o(N.log2N)

SYSTEM REQUIREMENTS

In the module implementation the following problems should be solved:

- sampling of a sound signal with the means of the multimedia system of Microsoft Windows with the possibility of managing the digitization parameters;
- saving the calculated values in temporary buffer and visualization of the signal wave form;
- execution of fast Fourier Transformation on the obtained values;
- visualization of the obtained spectrum;
- saving the data in a file.

For the realization of the program module for spectral analysis was not chosen the programming language C++, but the integrated development environment Visual C++. The reason for this is the fact that the C++ programming language, which guarantees high performance, is also suitable for realization of systems that require high speed of data processing for obtaining results close to the work in real time. Visual C++ is also one of the most common IDEs, which will help for wider application of the module.

ARCHITECTURE OF THE SOFTWARE SYSTEM

Fig. 1 shows the functional diagram of the programming module. As described an audio signal from a random source is connected to the input of the sound card. The restrictions for the parameters of the signal are determined from the technical characteristics of the sound card. The software digitizes the signal using the hardware characteristics of the audio card and the multimedia functions of MS Windows. The obtained values during sampling are saved in a cyclic buffer in the random access memory of the computer. This guarantees good performance of the software implementation. On basis of the saved data 3 main actions are executed – visualization of the input signal, fast Fourier transformation, visualization of the obtained spectrum and saving the data on the hard disk in case further processing is required. When realizing the described functions it is necessary to implement some level of parallelism of the processing, because the goal of module is the analysis and the visualization to be as close as possible to work in real time.

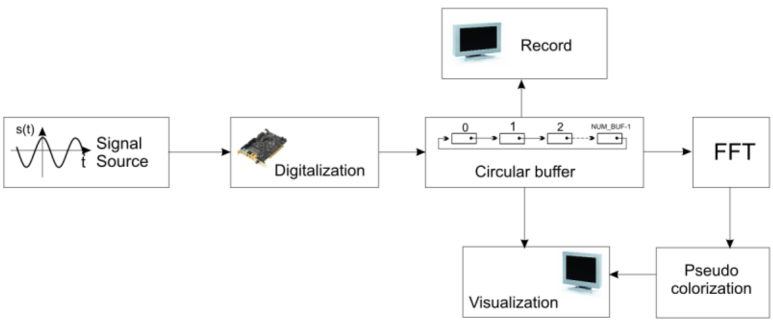

Fig. 1. Functional diagram of the software system

The software module uses the standard Windows multimedia system for working with the sound card. For processing the obtained digital data and their visualization suitable classes are implemented.

The problem of the visualization of the spectrum is more interesting. The result of the discrete Fourier transformation is a complex function, while for the visualization of the spectrum we get a three dimensional discrete function – the two axis display time and frequency and the third - the module of the calculated result. Because of the fact that this is difficult to implement programmatically, it is common to visualize only the spectrum at the current moment and in this way the function becomes two-dimensional: the X axis displays the frequency and the Y axis - the amplitude of the frequency components.

The developed module uses a visualization approach in which the function is displayed in the 2D space and for displaying the third dimension it uses a colour to encode the amplitude. Figure 2 illustrates the chosen approach, where a colour scheme with 4 colours is used.

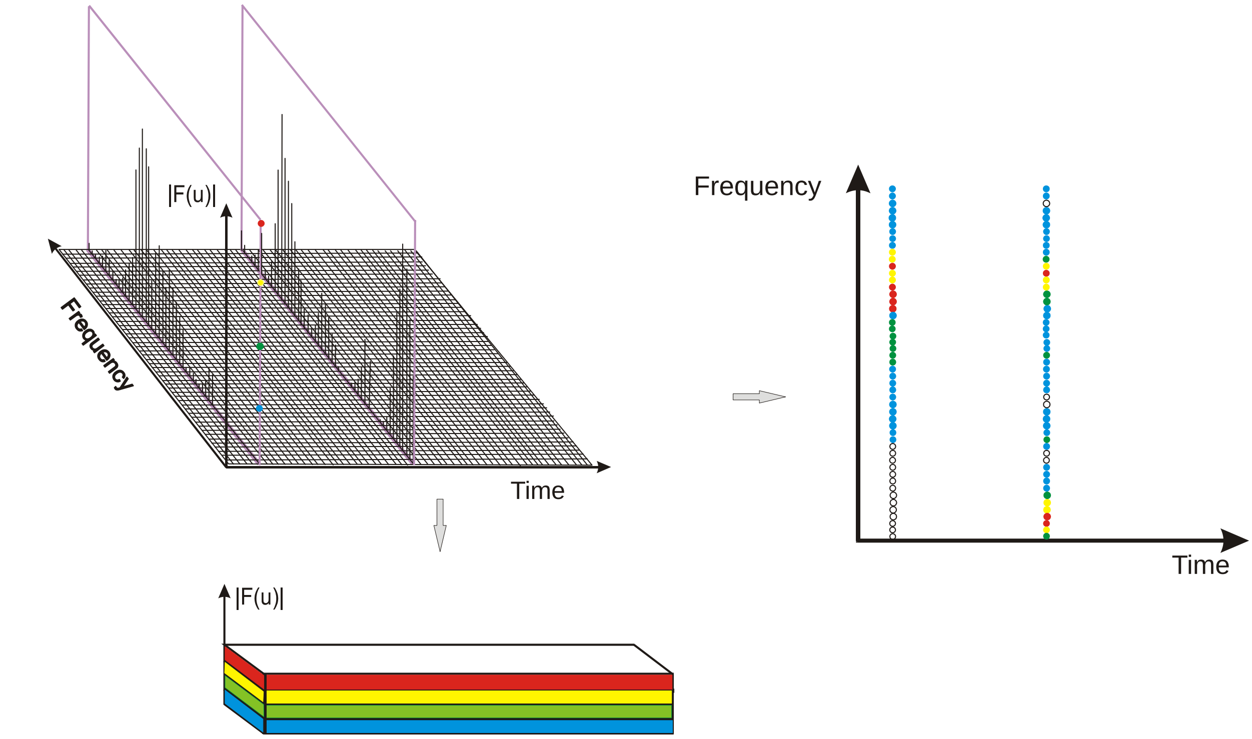

Fig. 2. A method of visualizing a frequency spectrum with colour encoding

In this way every point (x, y, z) is transformed to a point (x, y) for which the according colour is matched. Using this approach through a 2-dimensional image of the signal we have information not only for the spectrum in the current moment but also for the monitored period of time. After the display area is filled the program starts visualizing the data for a new time interval starting again from the left side of the zone. Using this method we lose the information of the previous time interval, that's why during the same time the spectrum is calculated and visualization is made, we also save the data in a binary file in order to be able to display them later or to analyse it with the help of other programs. For an accurate implementation the coding of the colour level with 4 colours is not sufficient. It is common to use colour schemes with 256 colours, which give more accurate visualization of the spectrum. It is also possible to use different colour schemes for the needs of the application. The presented implementation uses a colour scheme with 256 colours, where the cold colours correspond to small numbers and the warm colours - to big numbers. In practice in the two dimensional function can be seen a pixel with the corresponding colour.

The following approach is used for the implementation of the coloration in the developed system:

- 3 arrays with 256 elements are created, containing the RGB components of the corresponding colour;

```
static unsigned short
rr[256]={  0,  0,  0,  0,  0,  0,  0,....... 255,255,255},
gg[256]={  0,  0,  0,  0,  0,  0,  0,....... 7,  3,  0},
bb[256]={255,255,251,247,243,239,235,.......,0,  0,  0};
```

- The function MapColor returns as a result colour corresponding to the given value between 0 and 255;

```
COLORREF MapColor (int s){
    return RGB(rr[s],gg[s],bb[s]);
}
```

- The Fourier spectrum is visualized with the following code:

```
for (int i = 0;
     i < fftTransformer.Points() / 2 && i < rect.bottom;
     i++ ){
          int s = int (fftTransformer.GetIntensity(i) / 256);
          COLORREF color;
          if (s > 16){
               color = MapColor (s);
               canvas.Point (_xRecord, rect.bottom - i - 1, color);
               canvas.Point (_xRecord + 1, rect.bottom - i - 1,
               color);
          }
     }
_xRecord += 2; //increase X coordinate
```

RESULTS

The system was implemented under Windows using Microsoft VS. Fig. 3 shows the interface of the program, which implements the spectral analysis. It has controls which grant access to some of the parameters of digitization and Fourier transform – frequency of sampling, number of bits for quantization on a certain level, number of points for FFT. The frequency of sampling can be selected from the following values: 11025, 22050 и 44100 Hz. The amount of bits for the sampling on a certain level can be selected with a radio button. The program allows the user to choose from a drop list the number of dots for the fast Fourier transforms – 256, 512, 1024, 2048. There is an option to select the amount of transformation for a second using a slider.

For this particular application of the program it is enough to use sampling frequency of 22050 Hz. For a greater precision it is recommended to use 16 bits value for the transformation of the signal into a digital unit. The data cannot be processed in real time because of the software realization of the spectral analyser. Despite that it is guaranteed that the delay between the receiving of the signal and the visualization will be the same for the form and for the spectrum. In order to reduce this delay it is recommended for slower computers to use fewer transformations in seconds and fewer points for transformation.

The result of the digitization is a sequence of values. This sequence (form of the signal) is displayed by the program in the upper right area. The program displays the results of the measurement for a certain amount of time. After the reaching of the right end of the area the drawing starts again from the left to the right, as a vertical line moves, separating the previous interval from the new one. This principal is the same as the Doppler ultrasound device with built-in function for analysis.

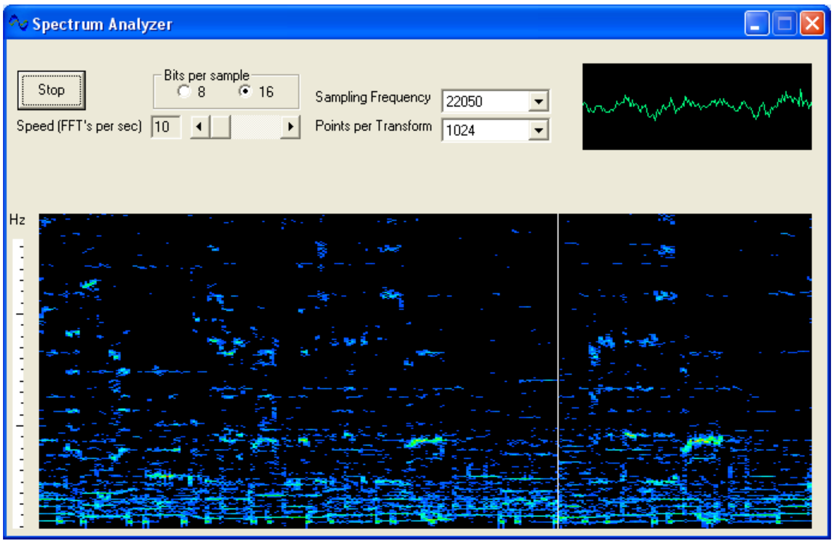

Fig. 3. Program for spectral analysis

The visualization of the form of the wave is not precise and there is no calibration option for the level of the signal. It is only used to evaluate the presence of a signal and its amplitude. The spectrum of the signal is important in this device and it is displayed in the lower rectangle. The encoding is done using the described scheme and due to the specification of the medical measurements the values of the module $|F(u)|<16$ are displayed in black. For greater values a 256 colour scheme is used. The colour scale is displayed on the right side of the area, and in the left there is a scale with the frequencies.

CONCLUSIONS AND RECOMMENDATIONS

The created software module allows the analysis of sound signals using universal computer. The digitization of the signal is performed by universal sound card and that is why the results are not precise, but the solution could easily be applied in many areas (fields) – sound reproduction, voice analysis, academic studies, Doppler ultrasonography, etc.

REFERENCES

[1]. Kruglinski D., G. Shepherd, S. Wingo, Programming Microsoft Visual C++ , Microsoft Press, 1999.
[2]. Sophocles J., Introduction to Signal Processing, Prentice Hall, 2010.
[3]. Smith S., The Scientist & Engineer's Guide to Digital Signal Processing, California Technical Publishing, 1997
[4]. Richard B., V.Lazzarini (Editor), M. Mathews, The Audio Programming Book, Massachusetts Institute of Technology, 2011
[5]. Oppenheim A., A. Willsky, (Author), S. Hamid, Signals and Systems (2nd Edition), Prentice Hall, 2003.

CONTACT ADDRESSES

Pr. Assist. Rumen Ivanov Rusev
Department of Informatics and IT, FNSE
Angel Kanchev University of Ruse
7017 Ruse, Studentska Str. 8, Bulgaria
Phone: (++35982) 888 326
Email: rir@uni-ruse.bg

Ana Kancheva Kaneva,
Department of Informatics and IT, FNSE
Angel Kanchev University of Ruse
7017 Ruse, Studentska Str. 8, Bulgaria
Phone: (++35982) 888 852
Email: akaneva@uni-ruse.bg

ISSN 1314-3077
9 771314 307000