БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ . BULGARIAN ACADEMY OF SCIENCES

ПРОБЛЕМИ НА ТЕХНИЧЕСКАТА КИБЕРНЕТИКА И РОБОТИКАТА, 49
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS, 49

София . 2000 . Sofia

A Software System for Visualization and Editing of Two-Dimensional Signals with Large Duration Designed for the Purposes of Digital Signal Processing (DSP)

Ivan Moustakerov

Institute of Information technologies, 1113 Sofia

Introduction

In research practice, especially the processing of voice signals often requires the operation with two-dimensional signals of large duration (dozens of seconds). The recording of such signals in a digital form creates files of Mbyte dimensions. It is often necessary to define which part is of interest and should be recorded and which one to ignore during the process of signal recording. A similar problem can be solved with the help of appropriate dynamic visualization of the signal recorded, editing and recording of some selected parts. Several problems appear when realizing these requirements.

Main problems

The first main problem is connected with the dimension of the file storing the signal in a digital form. The usual dimension of the file with the data is about 1–1.5 Mbytes. The limited dimension of the accessible conventional memory under DOS does not allow the data file to be loaded entirely in memory and hence it requires the use of extended memory type.

Another problem is the necessity for dynamic visualization of the signal, i.e., its indication, imitating the recording process. Signal visualization is required due to the need of editing, i.e., its displaying in arbitrary direction, entirely and in parts. The recording speed, which is usually achieved with the help of quick specialized digital processors excludes the possibility for visualization in “on-line” mode. This implies the recording of the entire signal at first, and later on, in “off-line” mode, the imitation of the signal recording process.

The rest of the problems that remain to be solved, are connected with the necessity to edit the file recorded – “cooling” and selection of signal parts, scaling (increasing or reducing), deletion and recording of the selected parts into another file.

The first approach in solving similar problems is the supply of appropriate software products, for example - MATLAB, MathWorks, DADISP, Entropic and others. However, their use has to account that in the general case these are integrated software systems, where visualization is just one of many other possibilities. This leads to great size of the products and their high price. The fact that their complete application usually requires considerable efforts for learning and getting acquainted with each product should not be underestimated too, since it supposes time and certain "software" disposition which is not always typical for the researchers.

Having in mind the above said, the respective software design can be accepted as appropriate in some cases of projects realization. In order to justify this approach, the corresponding software system must execute the necessary functions only, to be possibly smallest in volume and run on available computers, i.e., not setting specific requirements towards them. Similar considerations refer to the applicability of the widely spread Windows operating system. It exists in several different (and incompatible) versions – Windows 3.x, Windows 95, Windows NT, each of them implying some requirements with respect to the technical features of the computer used and needing specific (as well as expensive) set of software development tools. Taking into account these considerations, and in connection with the realization of a project for DSP, it has been decided to design a software system for dynamic visualization and processing of two-dimensional signals with large volume, operating under DOS, its description being an object of the present paper.

Algorithmic block-diagram of the software system

A modular structure of the system has been designed according to the method "up-down", consisting of the following modules, shown with their interconnections in Fig. 1:

– a control module;
– a module for reading/recording of the file with the signal data;
– a module for loading the signal in the operating memory available;
– a module for graphic dynamic indication of the signal;
– a module for signal editing;
– a module for "hard-copy" of the image on a printer;
– a module for errors processing (system and user's).

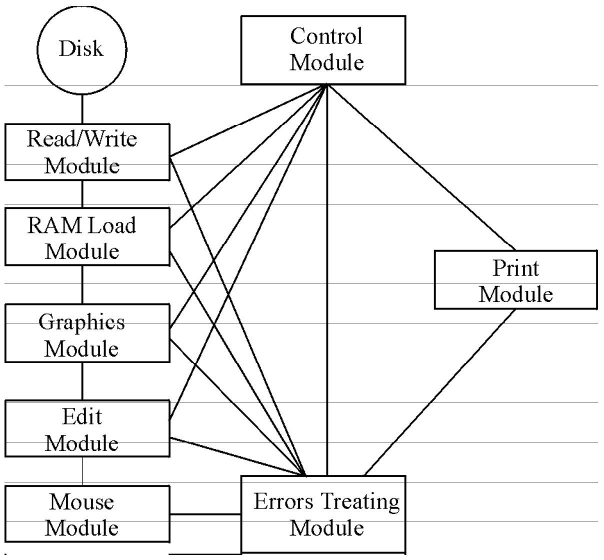

Fig. 1

Control module. It synchronizes the work of the remaining modules and transmits some of the data necessary for their operation. It indicates the menus on the screen, realizes a temporary output to DOS and return to the main graphical screen.

Read/write module. It reads and writes the file with the signal data from the available disk devices. It realizes the necessary interface with the user - questions for the file name, disk device, messages for missing and existing file, etc.

A module for loading into RAM It is the main module in solving the problem between the operating memory and the dimension of the data file. It defines the dimension of the available memory ("extended" memory included). It realizes the connection with XMS-memory from DOS environment. It sets the necessary indicators and transfers them to the control module, where they are accessible for all the remaining modules.

Graphics module. Also a main module, testing for an available videocontroller, for possible graphical modes and setting a mode with the maximum possible capacity. The module visualizes the signal dynamically and responds to the requests of the editing module and the indicating module by the "mouse" or the arrow keys.

Editing module. It realizes the basic interface with the user. With the help of the indicating module (by "mouse" or arrow keys) it stops the dynamic indication of the signal, selects certain parts of the signal in arbitrary direction (back or forward), scales these parts (increase or decrease), deletes them or transmits them for recording on the disk (by the read/write module).

"Mouse" module. It realizes direct selection of parts from the signal by the user. Checks for the presence of a "mouse" and makes it accessible for the user in case it exists, if there is not any, it realizes the same functions as the arrow keys.

Error processing module. The module is designed mainly for processing system errors of "Abort, Retry, Ignore" type, shown on the graphical screen and destroying the image. The module detects such errors, transmits them to the control module which on its turn shows them on an appropriate place on the screen.

Printing module. It prints (if there is a printer) the screen part selected by the user. It can create a print file for a definite type of a printer (if there is not any at the moment) for its transfer and printing on another computer. It supports printers of Epson type at the moment.

Program realization

A program language C^{+} has been used and Turbo C^{++} environment. The program system consists of the following files:

– VISIG.h – a heading file with definitions of the functions and constants used;
– GRASIG.C – a file, containing auxiliary graphical functions;
– VISIG.C – a main file, realizing the system algorithm.

GRASIG.C file contains the following functions:

```
static void restore_old_break(void)
static void interrupt newInt9(void)
static void installBreak(void)
void initgr(void)
void invertpixel(int X, int Y)
int IX(float x)
int IY(float y)
void move(float x, float y)
void draw(float x, float y)
void to_text(void)
void endgr(void)
void grtext(float xleft, float ylower, float heghtpercent, char *str)
void far *far _graphgetmem(unsigned size)
void far _graphfreemem(void far *ptr, unsigned size)
void line_uc(float x1, float y1, float x2, float y2)
void circle_uc(float x, float y, float r)
int XPIX (float xdim)
int YPIX (float ydim)
void arc_uc(float xC, float yC, float stangle, float endangle, float radius, int
nlinesegments)
float angle(float x, float y)
float drawarc3(float xA, float yA, float xB, float yB, float xC, float yC,
        float *pr)
float fillet(float xA, float yA, float xB, float yB, float xC, float yC,
        float r)
void circle80_pc(int XC, int YC, int R)
void circle80_uc(int xC, int yC, int r)
void fatline0(float x1, float y1, float x2, float y2, float d)
void fatline(float x1, float y1, float x2, float y2, float d)
void sharpjoint(float d)
```

VISUG.C file contains the following functions:

```
void main(void)
void circ_arc(int *px, int *py)
int Xpixel(int x), Ypixel(int y), xhpg(int X), yhpg(int Y)
int mousinit(int Xlo, int Xhi, int Ylo, int Yhi)
void mousget(int *pX, int *pY, int *pbuttons)
int mousread(int *pX, int *pY, int *pbuttons)
void cursor(int X, int Y)
void addline(int Xstart, int Ystart, int X, int Y)
void addstring(int X, int Y, char *str)
void wrlines(void)
int confirmed(void)
void rdlines(void)
void initmenu(void)
int getstring(int X, int Y, char *mes, char *str, int boxcode)
void clearrectangle(int Xtop, int Ytop, int Xbottom, int Ybottom)
char ermes(char *str)
void displaybotline(char *str)
void defaultbotline(void)
void drline(int x1, int y1, int x2, int y2)
void drawnewline(int x1, int y1, int x2, int y2)
void endprogram(void)
void newposition(int *px, int *py, int *pX, int *pY, int *pbut)

void refresh(int x, int y)
void textinput(int *px, int *py, int *pX, int *pY)
void selectobject(int *pX, int *pY)
void getline(int *pXstart, int *pYstart)
void txtcursor(int X, int Y)
void selectlinoptions(int i,int *pX,int *pY,int XX1,int YY1,int XX2,int YY2)
void selecttxtoptions(int i, int *px, int *py)
void secondmenu(void)
void boxes(int n, int primary)
void invertbox(int i, int x, int y, int primary)
void os_shell(int x, int y)
void arrowhead(int *px, int *py)
int round(float x)

#define txt(linenr, str) gotoxy(1,linenr) cprintf(str)
```

The software system is compiled in a model in HUGE memory.

Conclusion

The first version of the system is tested recently. The main modules are cleaned from errors and have proved their functionality. The connection with XMS memory is in the process of testing. The principal problems are solved, but the use of weakly documented DOS properties and possibilities requires considerable time for debugging and testing. The development of a version operating under Windows can be done after generalization of the results concerning the use and the necessity for such version of Windows.

References

1. IEEE Signal Processing Magazine. Periodicals, 1997, 1998.
2. P a p p a s, C. H., W. H. M u r r a y. Borland C^{++} Handbook. Osborne, McGraw-Hill, 1991.
3. L a m e r a l, L. Principles of Programming in Traffic. M., Sol-System, 1992.

Программная система визуализации и редактирования двумерных сигналов большой длительности для целей цифровой обработки сигналов (ЦОС)

Иван Мустакеров

Институт информационных технологий, 1113 София

(Р е з ю м е)

Работа связана с научно-приложной областью исследования речевых сигналов. Практика показывает, что она связана с трудностями при записи, визуализации и редактировании исследуемых сигналов. Предложен подход – разработка так называемого custom software для определенного программного продукта. В будущем будет представлена версия, работающая в среди Windows.